Description of Commands

General notes about the command descriptions

Declarations

(blank) Blank entry, no entry required, command parameter can be left out

int Decimal value, integer, absolute (integer)

num Decimal value, with following point positions, absolute (numerical)

name Text string which names a variable – the following special characters may not be part of the string:

Char.			!	"	#	,	.	+	-	*	/
Name		blank	exclam	quotedbl	numbersign	comma	period	plus	hyphen	asterisk	slash
ANSI	<32	32	33	34	35	44	46	43	45	42	47
Hex	<20	20	21	22	23	2c	2e	2b	2d	2a	2f

expression Can contain konstant values, variables and functions.

TEXT Text string, alphanumerical

# Commando character, designates a command

❍ For further information and command overview see topic section "Definitions and Command Overview".

❍ The printer groups can be found in topic section "General Information and Interfaces".

Variable Data Fields

If a variable data field (text field) is to be used, the variable data field is defined with the help of TEXT as follows:

TEXT = $n,c With:

$ Marker
n Number of the variable data field (0 to 99)
, Separator
c Number of characters in the variable data field (0 to 255)

Example	#YTz/0D///$n,c#G

❍ For an application example goto #YV.

Input Fields

(Only group H)
Input fields are not filled with content, until the print job is started. The input for each input field is queried only once, shortly after the print job was started. This input counts for all labels of the label amount designated in the print job.

TEXT = $<prompt>,default

With:

$ Marker
prompt Arbitrary text which shows up on the printer display together with the value *default*. This text allows to distinct several input fields. This text will not be printed.
, Separator
default Defines a default value for the respective field, which appears on the printer display together with *prompt*:

Example	#YT101////$<Price:>,10,- Euro#G

❍ For detailed information about input fields and the offline functionality of the printer, please refer to the User Manual 64–xx, AP 4.4/5.4 or DPM/ALX 92x, topic section "Advanced Application", chapter "Offline Function".

Immediate commands

#!A1 Interface activation

Definition Activates Easy Plug (after the device has been switched on or after it received a passive command #!P1).

Following command #!A1, all incoming data is evaluated as Easy Plug commands.

After the device has been switched on, the active command must be sent once, otherwise the printer will not work.

Syntax	#!A1	Groups A, B, D
	#!An	Groups E, G, H

#!A1 = Following this command the printer is in active mode, and commands which are received are carried out wherever possible.

n = int Set printer ID number (0 to 31).

#!CA Easy Plug - erase everything

Definition Resets Easy Plug to a predefined initial status:

- After a started label has been completed, an active printing process is stopped and broken off.
- The label format memory is erased.
- Material settings remain unchanged (width, length, reel/gap, material designation).
- The Easy Plug interface remains activated (that is, the effect of a preceding #!A1 command is maintained).

➡ But: During the act of erasing, for a short moment, no data can be transmitted (because the handshake lines are deactivated for an instance).

- Spooler (queue) is erased.
- The error number for the #!Xn response string is deleted, if the error is already acknowledged.

Syntax	#!CA	All groups

#!CA = Resets Easy Plug:
– Labels being printed are completed
– Series is broken off
– Spooler is erased
– Easy Plug waits for further formats

#!CF Erase format

Definition Breaks off the printing of the series currently being printed.

The current printing process is stopped after a label which has already been started is completed.

The label format memory is erased. Data then present in the spooler buffer is processed further.

Syntax	#!CF	All groups

#!CF = Breaks off the printing process:
– Labels being printed are completed
– Series is broken off
– Spooler data continues to be processed

#!D Trigger Single-Start

Definition The #!D command triggers exactly one Single-Start.

➡ Group H: Start signal input must be activated!

❍ For details about activating the start signal input read the user manual, topic section Advanced Applications 🗋, chapter „Printing with start signal", "Settings in the parameter menu".

➡ Group H: The command #!D can not be used as start signal in applicator operation mode, if the applicator is PLC-controlled (optional PLCs are available for LTSI, LTP and LTPV).

❍ Details about PLC-controlled applicators, see manuals LTSI 🗋 and LTP/LTPV 🗋.

Syntax	#!D	Gruppe H

#!Dn Single start function on/off

Definition The command #!D switches the single start function of the printer on/off.

Syntax	#!Dn	Group D (only TTX 300 Online, Cobra Online)

n = 0 Switch off single start
= 1 Switch on single start

The command is not saved. The printer is in standard mode after the device is switched off and then back on again, or after a reset.

Standard mode:

TTX 300 Online	= no single start
TTX 300 Dispenser	= single start

A = TTX 600 --TLX 604 – Eagle **B** = TTX 300 Offline - Cobra Offline **D** = TTX 300 Online –TTX 207 - Cobra Online – Da Capo
E = TTX 450/650/674/675/950/1050 – Puma/Puma plus/Lion/Lion plus/Tiger/Tiger XXL -- S45/65/95/105 - TDI/STDI/XXTREME
G = TTX 350 – Ocelot – TTK – Texxtile **H** = 64-xx – Chess x – DPM – PEM – ALX 92x – AP 4.4 – AP 5.4 – AP 5.6 – AP 7.t

#!H Head temperature adjustment

Definition Sets the head voltage and therefore the head temperature to the required value n.

Syntax	#!Hn	Groups E, G, H

n = int Head voltage (0 to 99).
0 = lowest head voltage, 99 = highest head voltage.

#!P1 Interface deactivation

Definition Switches Easy Plug into a passive state.

After receiving the command #!P1, all incoming commands are ignored with the exception of the active command #!A1.

After #P1 the active command must be sent once, otherwise the printer will not work.

Syntax	#!P1	Groups A, B, D
	#!Pn	Groups E, G, H

#!P1 = After receiving this command, the printer ignores all information received via the interface until the active command #!A1 is received.

n = int Set printer ID number (0 to 31).

#!PG Reading out Parameters

Definition Reads out parameter settings via the set interface.

The currently set parameter value is given back as ASCII text, which is terminated with <LF> (0x0a). Additionally, there is the option of reading out all parameters together. Those can then be transferred to another printer (see command #PC).

➡ In case of a not available ID, only <LF> is given back.

➡ Even those kind of parameters may be read out, which only appear under certain conditions in the menu.

Syntax	#!PG<ParameterID>#G	Group H

<ParameterID> Parameter ident number (ID). Each parameter in the menu has an own Parameter-ID.

Special ID	Function
-1	Printer specific settings (e.g. printhead resistance, sensor settings) are *commented out* (by a preceding "*"). By doing so, these settings are *not* transferred to another device (example 1).
-2	Printer specific settings (e.g. printhead resistance, sensor settings) are *not* commented out. By doing so, these settings are transferred together with the other settings.

A = TTX 600 --TLX 604 – Eagle **B** = TTX 300 Offline - Cobra Offline **D** = TTX 300 Online –TTX 207 - Cobra Online – Da Capo
E = TTX 450/650/674/675/950/1050 – Puma/Puma plus/Lion/Lion plus/Tiger/Tiger XXL -- S45/65/95/105 - TDI/STDI/XXTREME
G = TTX 350 – Ocelot – TTK – Texxtile **H** = 64-xx – Chess x – DPM – PEM – ALX 92x – AP 4.4 – AP 5.4 – AP 5.6 – AP 7.t

Special ID	Function
-4	Additional output of parameter names and values as comments. Furthermore, some printer settings are written at the beginning of the output file (example 2).
-8	Additional output of parameter type and setting range as comments (example 3). More information about parameter types can be found further below.
-16	Additional output of service data as comments (example 4).
-30	Combination of the values -16, -8, -4, -2 by addition: (-16)+(-8)+(-4)+(-2)=(-30). This activates all above mentioned types of comments (example 5).

Tab. 1 Parameter-IDs for special functions.

Example 1 Printer specific commands „commented out“	#!PG-1#G	#!A1 #PC1001/1 #PC1002/8 #PC1003/4.00 #PC1004/4.00 *PC1024/30 (← preceding “*”) …

Example 2 Additional comments	#!PG-4#G	#!A1 #G Machine Setup for AP 5.4 300 Dpi Version: R2.32P PE2.32C #G Serial number : 12345657890 #G MAC Address : 00.0a.44.02.09.89 #G Creation date : 16.06.2004 09:23 #G-- #G Printer Parameter Menu #G-- #PC1001/1 #G Infeed no.: Nr. 1 #PC1002/8 #G Inf. change spd. : 8 Inch/s #PC1003/4.00#G Print speed : 4 Inch/s #PC1004/4.00#G Feed speed : 4 Inch/s #PC1005/1 #G Materialtype : Punched #PC1006/35.00 #G Materiallength : 35.0 mm #PC1007/104.00 #G Materialwidth : 104.0 mm #PC1008/0.00#G Punch offset : 0.0 mm …

Example 3 Additional comments (setting range)	#!PG-8#G	#PC1505/4000 #G <i: 1024 - 65535 (1)> #PC1506/0 #G <d> #G 0 = Automatisch #G 1 = 10M Halbduplex #G 2 = 10M Vollduplex #G 3 = 100M Halbduplex #G 4 = 100M Vollduplex #PC1508/avery #G <s: 16> …

Example 4 Additional service data as comments	#!PG-16#G	… #G-- #G Netzteil Daten #G-- #G #G Typ : HME PSupply 450 #G #G Version : H0.40 F1.32 #G #G Serien Nummer : 02510012 …

Example 5 All types of comments	#!PG-30#G	#!A1 #G Maschinen Setup für AP 5.4 300 Dpi Version: R2.32P PE2.32C #G Serien Nummer : 12345657890 #G MAC Adresse : 00.0a.44.02.09.89 #G Erstellungsdatum : 16.06.2004 10:33 #G-- #G Drucker Parameter Menü #G-- #PC1001/1 #G Einzug-Nr. : Nr: 1 #G <i: 1 - 4 (1)> #PC1002/8 #G Einzugs. Geschw. : 8 Inch/s #G <i: 4 - 10 (1)> #PC1003/4.00#G Druckgeschwind. : 4 Inch/s #G <f: 2.0 - 6.0 (1.0)> #PC1004/4.00#G Vorschubgeschw. : 4 Inch/s #G <f: 2.0 - 6.0 (1.0)>

INFO: Parameter types

The printer provides 4 parameter types, whose setting range and assignment can be queried.

<i> Integer parameter (e.g. PRINTER PARAMETER > Bar code Multip.)
Integer value; must match the setting range.

Example	<i: 1024 - 65535 (1)>	Setting range: 1024-65535 Unit interval: 1

<f> Float parameter (e.g. PRINTER PARAMETER > Print speed)
Value with decimal places; must match the setting range.

Example	<f: 0.0 - 999.9 (0.1)>	Setting range: 0.0-999.9 Unit interval: 0.1

<s> String parameter (e.g. INTERF.PARAM. > ETHERNET PARAM. > FTP password)

Example	<s: 16>	Maximum string length: 16 characters

<d> Discrete parameter (e.g. SYSTEM PARAMETER > Periph. device)
Integer value with a special meaning.

Example	<d>	0 = RS232 1 = RS422 2 = RS485

#!SP Stop printing

Definition Stops the printing of a label series.

Printing of a label is completed. This command is ignored if the printer is already on stand-by (finished or no #Q). Data from the spooler (queue) is not evaluated. Only immediate commands are processed.

Syntax	#!SP	All groups

#!SP = Stops the printing:

- Labels being printed are completed
- Series is halted
- Spooler is no longer processed
- Printing process only restarts after the printer receives a start command #!SR.

#!SR Start printing

Definition Starts the printing of a halted series.

The remaining labels of a format which has been halted by the stop command #!SP are printed.

Syntax	#!SR	All groups

#!SR = A series halted by a stop command is restarted. The remaining labels of a halted format are printed.

#!X Status acknowledgement V.24

Definition Request for a status acknowledgement via V.24.

Syntax	#!X	Group A

#!X = Status request:

- acknowledgement only possible only with a V.24 (TTY) interface (CTS pin on TRUE)
- the acknowledgement is given with the same parameters as the receive channel (same Baud rate, parity, number of stop bits, data bits)
- acknowledgement takes place without a handshake and inter-character delay

Acknowledgement (1 byte) as follows:

7	6	5	4	3	2	1	0

0	= 0	Stacker OK
	= 1	Stacker full
1	= 0	Material OK
	= 1	Material end
2	= 0	Ribbon OK
	= 1	Ribbon end
3	= 0	Head contact pressure OK
	= 1	Head open
4	= 0	Quantity 0 (zero)
	= 1	Quantity greater than 0 (zero)
5	= 0	Interface OK
	= 1	Error during transfer
6	= 1	Always = 1
7	= 0	No other errors
	= 1	Other errors

#!XB Status ackn. with bar code

Status acknowledgement with bar code

Definition

After a label has been printed, #!XB delivers the printer status and the bar code on the label. Delivery normally occurs in 14 Bytes.

➡ Exception: Delivery of 26 Bytes, if:
a) short material (ABO or MFK) is being used, and
b) material end is reached
in this case, the bar code of the last label and the label before are delivered.

Syntax #!XB Group H (only TT4)

Bit Nr.:	7	6	5	4	3	2	1	0
Byte 0:	Printer type: 2 = TT4				No. of last supply unit to transport material to the front.			
Byte 1:	Last printer error message: 0000 0101 bin = 5 dez Material end 0000 0111 bin = 7 dez Ribbon end 0110 0011 bin = 99 dez other errors							
Bytes 2 bis 13:	12-figure bar code of the last printed label							

Special case: Delivery of 26 Bytes:

Bit Nr.:	7	6	5	4	3	2	1	0
Byte 0:	see above							
Byte 1:	see above							
Bytes 2 to 13:	12-figure bar code of the last regular printed label – that is one before the very last label of the roll.							
Bytes 14 to 25:	12-figure bar code of the last label – which is crosshatched.							

#!XC Pharmacy Code ackn.

Pharmacy Code acknowledgement

Definition #!XC delivers the data of the last printed pharmacy bar code.
Delivery occurs in the form:

xxxxxxxyyyyyyyy (Code content without checksum), with:

xxxxxxx = 7-digit CNK number
yyyyyyyy = 8-digit sequential number

If no pharmacy bar code was printed since the last printer reset, the delivery is 000000000000000.

Syntax	#!XC	Groups E, H

#!Xn Bar code acknowledgement

Definition #!Xn delivers bar code.

Syntax	#!Xn	Group H (only TT4)

#!Xn =	Request for bar code
n = 0	Delivers all bar codes: Four 12-figure bar codes = 48 bytes
n = 1	delivers the bar code of the 1st supply unit (12 bytes)
n = 2	delivers the bar code of the 2nd supply unit (12 bytes)
n = 3	delivers the bar code of the 3rd supply unit (12 bytes)
n = 4	delivers the bar code of the 4th supply unit (12 bytes)

#!Xn Status acknowledgement

Definition

Groups B to G:

Request for a status acknowledgement via RS232 or RS485/422.

Group H:

Request for a status acknowledgement using the selected interface. For transmission via Centronics, nibble mode is used.

All groups:

➡ Notes concerning status acknowledgement via serial interface:

– Acknowledgement at RS232 will only work, if the CTS pin is set to TRUE.
– The acknowledgement is given with the same parameters as the receive channel (same Baud rate, parity, number of stop bits, data bits).
– Acknowledgement takes place without a handshake and inter-character delay
– A delay period can be set.
– The status number is maintained in the acknowledgement string until the status changes or a reset or the command #!CF is performed.

Syntax	#!Xn	Groups B, D, E, G, H

n =	int	Acknowledgement delay period
=	0...9	0 = 0 ms, 1 = 100 ms, 2 = 200 ms etc. up to 9 = 900 ms

Acknowledgement string
(groups B to G) **S**aa**A**bcd**M**qqqqq**F**eeeee**K**xx

➡ The acknowledgement string is always 22 characters long (in ASCII, decimal digits, without spaces).

a =		Last status number (as shown in operating panel display), 00 = no error
b =	0	If status has already been requested with #!Xn
=	1	If status is being requested for the first time
c =	0	If status has already been acknowledged at the operating panel
=	1	If status is displayed, but has not yet been acknowledged
d =	0	If there is no label format stored
=	1	If a format has been opened, but not closed with #Q
=	2	If a valid format is being printed
q =		Number of labels still to be printed
e =		Number of bytes still available in the spooler buffer
x =		EPROM version number (e.g. 10 = Version 1.0)

Example	#!X0	Status request with 0 ms delay period. Example acknowledgement: S00A000M00054F12000K10

Acknowledgement string

(Gruppe H) **S**aaaa**A**bcd**M**qqqqqq**F**eeeeee**K**xxxxxxxxxxxxxxxx

➡ The acknowledgement string for group H printers is always 40 characters long (in ASCII, decimal digits, without spaces).

a	=		Last status number (as shown in operating panel display), 0000 = no error
b	=	0	If status has already been requested with #!Xn
	=	1	If status is being requested for the first time
c	=	0	If status has already been acknowledged at the operating panel
	=	1	If status is displayed, but has not yet been acknowledged
d	=	0	If there is no label format stored
	=	1	If a format has been opened, but not closed with #Q
	=	2	If a valid format is being printed
q	=		Number of labels still to be printed
e	=		Number of bytes still available in the spooler buffer
x	=		Firmware version no. – same text string appears on the lower display line shortly after switching on the printer.

Example	#!X0	Status request with 0 ms delay period. Example acknowledgement: S0000A100M000000F065536K R2.02 H2.01

#!XMn Diagnose dump / Read maschine status

Definition Reading out different machine states or Trigger Diagnose Dump and read out diagnostic data.

Return format: Value + linefeed

❍ Diagnose dump: See printer service manual, topic section „Fault location“, chapter „Reading out diagnostic data“.

➠ It is mandatory to finish the command with #G.

Syntax #!XMn Group H

n = -99 Diagnose dump is sent to the debug interface (which normally is Com1) and is stored in the flash memory.

n = -100 Diagnose dump is sent to the active data interface (INTERFACE PARA > EASYPLUGINTERPR > Interface) and is *not* stored in the flash memory.

n = 1004 Return value: 0 or 1
0 = Online
1 = Offline

n = 1007 Return value: number of printjobs, which are ready to print

n = 1008 Return value: 0 or 1
0 = Easy-Plug interpreter is *not* active
1 = Easy-Plug interpreter is active

n = 1009 Return value: rest print amount of the active printjob (equals the value returned by #!Xn)

n = 1010 Return value: 0 or 1
0 = An error message is currently displayed
1 = No error message

n = 1011 Return value: last or currently displayed error status number. Additional information about which of the two cases occurred gives #!XM1010

n = 1013 On-/Offline state; Return value:
0: Offline
1: Online
2: Online stopped

n = 1015 Printer activity; Return value: 0 or 1
0 = Printer is idle
1 = Printer is busy (printjob is being processed or printed)

n = 1201 Return value: Displayed text upper display line

n = 1202 Return value: Displayed text lower display line

Example	#!XM1004#G	“1” followed by “linefeed” is sent back to the host, if the printer is offline

#!XP Parameter status

Definition #!XP delivers a selection of parameter settings of the printer packed in one answer string.

- The answer string is 182 bytes long and contains all parameter settings listed in the table below in the same order.
- All parameter settings contained in the answer string are lined up without a separating character.
- Digits which are not used (see column no. of digits) are filled by preceding blanks.

Syntax #!XP Group H (only TT4)

Parameter	No. of digits	Example	Allocation / Unit
Version number	16	„R1.10 HS 0.15„	
Date	12	„Jun 15 2000“	
Printer type	16	„TT4„	
Infeed no.	1	„1“	
Inf. change spd.	2	„8“	Inch/s
Print speed	2	„11“	Inch/s
Feed speed	2	„12“	Inch/s
Material type	1	„0“	0 = endless 1 = punched
Material length	6	„175.4“	mm
Material width	6	„99.0“	mm
Punch offset	4	„-2“ „13“	mm
Bar code Multip.	2	„1“	
Cut mode	1	„0“	0 = Real 1:1 1 = Batch Mode 2 = Normal 1:1
Cut speed	2	„3“	Inch/s
Cut position	6	„2.0“	mm
Double cut	6	„10.0“	mm
UPC decoded line	1	„0“	0 = In line 1 = Raised
X – print offset	6	„-1.0“	mm
Y – print offset	6	„0.0“	mm
Punch mode	1	„0“	0 = Automatic 1 = Manuel
Punch level	3	„128“	
Interface	1	„0“	0 = RS232 1 = Centronics 2 = RS422 3 = RS485

Parameter	No. of digits	Example	Allocation / Unit
Baud rate	1	„2“	0 = 300 1 = 600 2 = 1200 3 = 2400 4 = 4800 5 = 9600 6 = 19200 7 = 38400 8 = 115200
No. of data bits	1	„8“	
Parity	1	„0“	0 = Odd 1 = Even 2 = Non 3 = Always zero
Stop bits	1	„2“	
Data synch.	1	„0“	0 = RTS/CTS 1 = XON/XOFF
Spooler mode	1	„0“	0 = Mult. print jobs 1 = Single print job
Printer ID no.	2	„1“	
Spooler size	3	„64“	KByte
w/wo magazine	1	„1“	0 = with 1 = without
Print interpreter	1	„0“	0 = Easyplug 1 = Lineprinter 2 = Hexdump
Character sets	1	„9“	0 = USA 1 = England 2 = France 3 = Germany 4 = Italy 5 = Swedem 6 = Spain 7 = Norway 8 = Special function 9 = IBM
Character filter	1	„0“	0 = Character > 20 H. 1 = All characters
Light sens. type	1	„0“	0 = Punched 1 = Reflex 2 = Full Size
Sens. punch LS	3	„100“	%
Ribbon autoecon.	1	„0“	0 = Deactivated 1 = Activated 2 = Thermal printing
Ribbon eco. limit	6	„10.0“	mm

Parameter	No. of digits	Example	Allocation / Unit
Turn on mode	1	„0“	0 = Offline 1 = Online
Single job mode	1	„0“	0 = Deactivated 1 = Activated
Head resistance	4	„1244“	Ohm
Temp. reduction	2	„50“	%
Voltage offset	3	„10“ ,“-20“	%
Singlestartquant	2	„1“	
Punch search qt.	2	„3“	
Mat. end detect	1	„0“	0 = Deactivated 1 = Activated
Periph device	1	„1“	0 = None 1 = Cutter 2 = Rewinder 3 = Tear-off edge 4 = Dispenser
External signal	1	„0“	0 = Disabled 1 = Single start 2 = Stacker full
Signal edge	1	„0“	0 = Falling edge 1 = Rising edge
Print contrast	3	„65“	%
Ram disc size	4	„128“	KByte
Font downl. area	4	„256“	Kbyte
Language	1	„0“	0 = German 1 = English 2 = French
Signal / buzzer	1	„1“	0 = Off 1 = On
Access authoriz.	1	„0“	0 = Deactivated 1 = Power-up code 2 = User 3 = Supervisor
Monitor version	10	„2.2a“	
Driver version	10	„1.0.2.3“	
Number of digits:	**182**		

#!XS Printer status acknowledgement

Definition #!XS reports the printer status.

Syntax #!XS Group H (only TT4)

#!XS = Request for printer status, delivered in 2 bytes:

Bit Nr.:	7	6	5	4	3	2	1	0
Byte 0:	Printer type: 2 = TT4				No. of last supply unit to transport material to the front.			
Byte 1:	Last printer error message: 0000 0101 bin = 5 dez Material end 0000 0111 bin = 7 dez Ribbon end 0110 0011 bin = 99 dez other errors							

B

#BCH Bar code height factor

Definition This command is used for magnifying the height of the bar code.

The factor relates to the height defined in the command #YB. This command may only be sent outside of a format #ER to #Q/.

Syntax	#BCHn	Group D

n = int Bar code height factor (1 to 10 Dot).

Default value is "1". The bar code height factor is stored in the NOVRAM.

Example	#BCH5 #ER ... #YB0/0/9////12345678	A bar code EAN 8 10 mm high multiplied by the factor "5" is transformed into a bar code 50 mm high.

#BOF Spooler buffer off

Definition This command is used to block the interface spooler buffer. The spooler buffer is blocked after receiving a format, and is only released after the required number of labels has been printed.

The #BOF command is not saved in Groups A and B, i.e. this command only remains active until the printer has been switched off or until a #BON command is sent.

➡ The command must stand outside of the command sequence #ER to #Q!

Syntax	#BOF	All groups

#BOF = Spooler buffer works in single job mode. Only one format can be sent.

#BON Spooler buffer on

Definition This command is used to activate the interface spooler buffer. It releases the spooler buffer after it has been blocked (with #BOF).

In this mode several formats can be sent to the spooler (even during the printing phase).

In Groups A and B the printer is always in BON mode after it is switched on.

➡ The command must stand outside of the command sequence #ER to #Q!

Syntax	#BON	All groups

#BON = Spooler buffer works in multi-job mode. Formats can be sent at all times.

#BR Break print job

Definition

With the command #BR, the printer stops in the format which follows the command #BR. The command #BR must be used outside of a format.

With #BR, the printer goes into stop mode and is restarted by pressing the FEED button.

➡ Command must stand outside of #ER and #Q!

➡ The parameter SMOD must be deactivated in order to be able to use the #BR command sensibly.

Syntax #BR Groups E, G, H

Example 1
The printer stops before the print job

```
#!A1
#G --------------------------------------
#G Printer goes into stop mode.
#G --------------------------------------
#BR
#IMN100/20
#ERN0//
#T10#J5#YT108/0///TEXT
#Q3/
```

Example 2
The printer stops after the first print job

```
#G --------------------------------------
#G The first job is printed immediately.
#G --------------------------------------
#!A1
#IMN100/20
#ERN0//
#T10#J5#YT108/0///
#Q3/
#G --------------------------------------
#G Printer goes into stop mode.
#G --------------------------------------
#BR
#!A1
#IMN100/20
#ERN0//
#T10#J5#YT108/0///
#Q3/
#BR
```

C

#CBF Bar code Codablock F

Bar code Codablock F

Definition The Easy Plug command #YB for bar codes is replaced by the special command #CBF for the Codablock F.

➡ Command must stand between #ER and #Q!

Syntax #CBF/dw/s/m/c/r/vo/a/TEXT#G Group H

d = 0 Bar code in normal writing direction
= 1 Bar code rotation 90 degrees
= 2 Bar code rotation 180 degrees
= 3 Bar code rotation 270 degrees

w = W The counting field TEXT is incremented/decremented without carry, that is, only the units position is increased/decreased.

s = int Bar code width (1 to 30); Default setting: 1

m = int Height of a Codablock-row in mm (1 to 100); Default setting: 5 mm

c = int No. of columns (4 to 62); Default setting: 10

r = int No. of rows (2 to 44); Default setting: 0

0: Number of rows is calculated by codablock.
1: Improper value, interpretation causes an error message.

v = + Increment – offset is added to TEXT
= - Decrement – offset is subtracted of TEXT

o = int Offset, which is added to or subtracted of TEXT, depending on the leading sign

a = int No. of labels with constant No. (1 to 255)

TEXT = Any alphanumerical text not exceeding the max. string length of 1024 characters. Consider the limitations of the bar code type.

➡ (Only group H) The text field may contain an *input field*.

❍ Syntax description: See chapter Input Fields 🗋 on page 4.

#G = The command #CBF must be closed with #G.

#CD Material change

Definition Command #CD switches to the defined material depot n. n is the supply chute of the change module, whose material is to be transported to the print head.

Syntax	#CDn	Group H (only TT2/TT4)

n = 1 1st Supply unit (top supply unit)
- not on TT2
- corresponds on the TT4 to the feed from the roll (TT4 parameter M_ROL)

= 2 2nd supply unit
- corresponds on the TT2 to the feed from the roll (TT2 parameter ROL)
- corresponds on the TT4 to the feed from the bottom magazine chute (TT4 parameter D_LEV)

= 3 3rd supply unit
- not on TT2
- corresponds on the TT4 to the feed from the middle magazine chute (TT4 parameter C_LEV)

= 4 4th supply unit (bottom supply unit)
- corresponds on the TT2 to the manual single label feed (TT2 parameter MAN)
- corresponds on the TT4 to the feed from the top magazine chute (TT4 parameter U_LEV)

#CF Delete file

Definition Deletes a file on RAM disk or CF-card.

➠ Command must be positioned *outside of* #ER and #Q.

Syntax	#CF/f	Group H

f = Filename, which is supposed to be deleted. The filename must be according to the DOS name convention (drive, path, name, extension).

➠ Always write file names in *capital letters*!

❍ For details about admissible file names read topic section "Notes, Definitions, Command Overview", chapter "Number string and text definition" / "File name conventions" and "Drive Names".

Example	#CF/A:\FONTS\FONT222.AFF#G	The file “FONT222.AFF” is to be deleted from RAM disk.

#CFN Code 49

Definition Uniform Symbology Specification Code 49 (ANSI/AIM-BC6-2000)

Syntax #CFNm/dkx/h/s/vo/a/TEXT#G Group H

m	=	0	Alphanumeric Mode
	=	1	Append Mode
	=	2	Numeric Mode
	=	3	Group Alphanumeric Mode
	=	4	Alphanumeric Mode, Shift 1
	=	5	Alphanumeric Mode, Shift 2
	=	6	Reserved
	=	7	Automatic Mode (default setting). The printer determines starting mode and encodation method by analyzing TEXT. This is the recommend mode.
d	=	0	Bar code in normal write direction
	=	1	Bar code rotated by 90 degrees
	=	2	Bar code rotated by 180 degrees
	=	3	Bar code rotated by 270 degrees
k	=	M	Bar code with plain copy line
		➠	The plain copy line can extendet beyond the right edge of the code!
	=	O	Bar code without plain copy line (default setting)
x	=	J	Plain copy line below the bar code
	=	A	Plain copy line above the bar code
h	=	int	Row height Row height = (h + 1) * PRINT PARAMETERS > Bar code Multip.
s	=	int	Bar code width factor (1 to 30 Dot)
v	=	+	Incrementation - offset is added to the TEXT field.
	=	-	Decrementation - offset is subtracted to the TEXT field.
o	=	int	Offset, which is added to (incremented) or subtracted from (decremented) the TEXT depending on the prefix.
a	=	int	Number of labels with the same constant number (1 to 255).
TEXT	=		Any alphanumerical text. The stipulations for the respective bar code must be taken into consideration. Max. number of characters: 255.

Accessing FNC Codes in “Automatic Mode”:

“<FNC1>”, “<FNC2>” and “<FNC3>” can be placed anywhere in the user data to insert the corresponding FNC code into the bar code.

❍ See examples below.

➽

Using special characters if not working with “Automatic Mode”:

➡ The user is responsible for the correct application of these special characters. Detailed knowledge about the code 49 is necessary.

❍ For detailed information about code 49 refer to ANSI/AIM-BC6-2000 “Uniform Symbology Specification Code 49”.

‘<’	S1	Shift1
‘>’	S2	Shift2
‘:’	FNC1	Function 1
‘;’	FNC2	Function 2
‘?’	FNC3	Function 3
‘=’	NS	Numeric Shift

Example 1
Automatic mode

```
#!A1
#IMN100.0/60.0
#N9
#ERN0///0
#T10#J4.5#CFN/M0/6/4///<FNC1>12345<FNC3>ABCDE#G
#Q1#G
#ERN0///0
#T7.5#J4.5#CFN/M0/6/4///MULTIPLE ROWS IN CODE 49#G
#Q1#G
#ERN0///0
#T7.5#J4.5#CFN/M0/6/4/1/1/EXAMPLE 2#G
#Q2#G
```

Example 2
Manual starting mode

```
#!A1
#IMN100.0/100.0
#N9
#PO0
#ERN0///0
#T15#J50#CFN2/M0A/4/4///12345=>ABCDEF<S>S#G
#Q1#G
```

#CG Adjust intercharacter gap

Definition

Adjust the intercharacter gap.

Group E only:

➡ This command can be used to alter the space between printed characters for fixed height fonts (command #YT). The space between characters can be reduced or expanded. The gap for scalable fonts (command #YT) cannot be changed.

➡ Only implemented in the US version of the firmware!
(from version 1U42 R11 on).

Both groups:

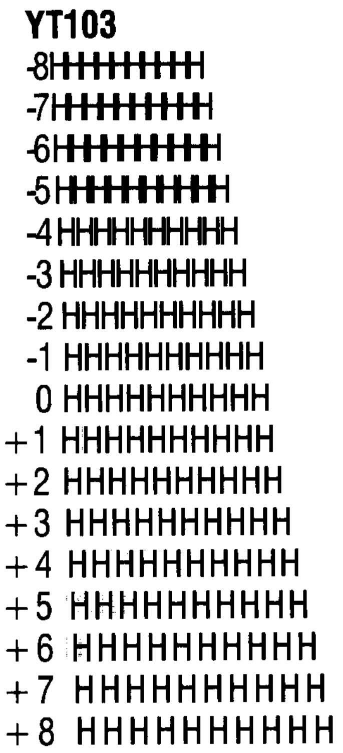
➡ The default size of the character gap depends on the used font size.

Syntax	#CGvn	Group E, H

v = + Enlarge the intercharacter gap compared with the default value.
= - Reduce the intercharacter gap compared with the default value.

n = int Number of dots to adjust gap (-8 to +8).
= 0 Sets the gap back to the default value.

Example

Data stream	**Sample Output**
#!A1#IMS82/180#ERN #J175.0#T035#YT106/0///YT103 #J170.0#**CG-8**#T035#YT103/0///-8 HHHHHHHHHH#**CG0** #J165.0#**CG-7**#T035#YT103/0///-7 HHHHHHHHHH#**CG0** #J160.0#**CG-6**#T035#YT103/0///-6 HHHHHHHHHH#**CG0** #J155.0#**CG-5**#T035#YT103/0///-5 HHHHHHHHHH#**CG0** #J150.0#**CG-4**#T035#YT103/0///-4 HHHHHHHHHH#**CG0** #J145.0#**CG-3**#T035#YT103/0///-3 HHHHHHHHHH#**CG0** #J140.0#**CG-2**#T035#YT103/0///-2 HHHHHHHHHH#**CG0** #J135.0#**CG-1**#T035#YT103/0///-1 HHHHHHHHHH#**CG0** #J130.0#**CGO**#T035#YT103/0///0 HHHHHHHHHH#**CG0** #J125.0#**CG+1**#T032#YT103/0///+1 HHHHHHHHHH#**CG0** #J120.0#**CG+2**#T032#YT103/0///+2 HHHHHHHHHH#**CG0** #J115.0#**CG+3**#T032#YT103/0///+3 HHHHHHHHHH#**CG0** #J110.0#**CG+4**#T032#YT103/0///+4 HHHHHHHHHH#**CG0** #J105.0#**CG+5**#T032#YT103/0///+5 HHHHHHHHHH#**CG0** #J100.0#**CG+6**#T032#YT103/0///+6 HHHHHHHHHH#**CG0** #J095.0#**CG+7**#T032#YT103/0///+7 HHHHHHHHHH#**CG0** #J090.0#**CG+8**#T032#YT103/0///+8 HHHHHHHHHH#**CG0** #Q1/	 YT103 -8HHHHHHHHHH -7HHHHHHHHHH -6HHHHHHHHHH -5HHHHHHHHHH -4HHHHHHHHHH -3 HHHHHHHHHH -2 HHHHHHHHHH -1 HHHHHHHHHH 0 HHHHHHHHHH +1 HHHHHHHHHH +2 HHHHHHHHHH +3 HHHHHHHHHH +4 HHHHHHHHHH +5 HHHHHHHHHH +6 HHHHHHHHHH +7 HHHHHHHHHH +8 HHHHHHHHHH

#CIM Cut

Definition Triggers the cutter or cutter-stacker, if available.

➡ The command must stand outside of the command sequence #ER to #Q, what means that it must stand outside of a print job!
➡ The command does not work in label formats for standalone mode!
➡ Group G: Set parameter PRTP > CLST to “Yes”, if the label material should automatically be fed forward and cut after the last label.
➡ AP 7.t: The #CIM command doesn´t work. Cuts have to be triggered using the #ER command.
➡ Printer with RFID option: The #CIM command doesn´t work. Cuts have to be triggered using the #ER command.

Syntax #CIM Groups A, E, G, H without AP 7.t

#CIM = Cut command.

Groups A, E, H:
Command placed after a print job: The label material is fed forward, is cut behind the last label and is fed backwards to the print position.

Group G:
Command placed after a print job: The cut is *not done immediately*, but is stored in the cut memory. Because of this group´s printers not being able to feed the material backwards, the cut is “saved” until the last label has reached the cut position. This may be the case after the printing of the next print job was started.

#CP No. of labels per colour field

Definition

The number of labels which are to be printed per colour field can be defined here. Varying quantities can be printed depending on the ribbon economy mode. The quantity can therefore be set by the operator.

The #CP command is positioned between the #IM command and the #ER command.

➠ The command must stand outside of the command sequence #ER to #Q!

Syntax	#CPa/b/	Groups E, F

a = int	No. of different colours on the ribbon
b = int	No. of labels which are to be printed with a colour field set

Calculating b

If the printer is not in ribbon economy mode, b is calculated as follows:

b = ribbon colour field length: label length (b integer)

Example:
colour field length = 100 mm, label length =20 mm, therefore b = 5

If the entire surface of the label is not printed, even more labels can generally be printed per colour field set in ribbon economy mode.

Consequently b is therefore dependent on
- the number of labels which are to be printed,
- the print position,
- the print speed and
- whether the labels are to be cut.

D

#DC Deleting all download logos

Definition Deletion of all stored logos.

This command deletes all the logos present in the download memory and releases the entire memory capacity for use.

➠ The command must stand outside of the command sequence #ER to #Q!

Syntax	#DC	Groups A-G
	#DCm	Group H

#DC = Every logo is deleted (without consultation) and the entire memory capacity is made available again.

m = analog to m = A.
= A Deletes every logo on the internal RAM disk.
= C Deletes every logo, which is stored on CompactFlash card in the folder “\logos”, if the file format is EPT.

➠ Logos which are not saved in file format EPT, but e.g. in JPG or BMP, are not deleted by this command.

#DF Downloading a file

Definition Downloads a file from a PC to the printer.

➠ To include the binary data of the file to be transmitted (see parameter b), you will need the help program „make_df.exe“.

make_df is contained on the Documentation-CD in folder „\utilities“.

❍ Notes about the application of “make_df” see topic section “Special Applications”, chapter Using MAKE_DF 🗋.

Syntax	#DF/s/f/i/b#G	Group H

s = same as s = N
= N Don´t overwrite file, if a file with the specified name already exists (default setting).
= O Overwrite file.

f = Filename, under which the downloaded data are saved. The filename must be according to the DOS name convention (drive, path, name, extension).

➠ Always write file names in *capital letters*!

❍ For details about admissible file names read topic section "Notes, Definitions, Command Overview", chapter "Number string and text definition" / "File name conventions" and "Drive Names".

i = int File size in bytes (1 bis 2^{32})

b = Binary file data

➽

#DK Downloading a logo

Definition

This command is used to download a logo (to send a logo from the PC to the printer) under a reference number (0 to max. 255) which must be entered.

Groups A to G (Group D without TTX 207):
The logo is stored on an initialised RAM card (if available) or in the logo buffer of the printer.

Group H:
The logo is stored on the internal printer RAM disk or on a – plugged-in – CompactFlash card.

➠ The command must stand outside of the command sequence #ER to #Q!

Syntax		
	#DKn/s/s ... /s#G	Group A
	#DKn//s/s ... /s#G	Groups B, D (without TTX 207), E, G
	#DKn/m/s/s ... /s#G	Group H

n = int — Logo reference number (0 to 255)

If the reference number has already been assigned to another logo or the memory capacity is insufficient
– in Groups B and D an error message is issued
– in Group A the transmitted logo is suppressed without an error message.

m = — analog to m = A.
= A — The logo is copied on the internal RAM disk (default setting).
= C — The logo is copied on CompactFlash card.
(in directory "\logos", respectively)

s = hex — Coding of a dot line in the logo matrix, hexadecimal with respectively 4 dots from left to right (000 to FFF).

Assignment for hexadecimal coding:
1 = dot is printed
0 = dot is not printed

Non-set dots at the end of the line can be left out.

Every line in the logo matrix is coded by one or more parameters per line, from bottom (Line 1) to top.

➠ Only capital letters and numbers may be used for hex. coding.

❍ For more information about creating a logo, see topic section “Special Applications”, chapter Using logos 🗋.

#G = — The command #DK must be closed with #G.

➽

Example (for groups B to G)	#DK1/C03/E07/F0F/FFF#G	Logo number 1 with subsequent creation is sent and stored in the download buffer. Line 4: 1111 1111 1111 (/FFF) Line 3: 1111 0000 1111 (/F0F) Line 2: 1110 0000 0111 (/E07) Line 1: 1100 0000 0011 (/C03)

#DM Download the name of a month

Definition This command is for downloading any name of a month.

The operator-specific names of months are entered one after the other, each being separated by a control character (^).

➠ The command must stand outside of the command sequence #ER to #Q!

Syntax #DM^ TEXT^ TEXT All groups

TEXT = Name of a month, freely definable, maximum 20 characters.

^ = Control character (^) for separating the individual month names.

Assignment:
The TEXT after the first control character (^) is assigned to the month of January, the TEXT after the second control character (^) to February, etc.

If less than 12 months have been defined, access to a missing month is ignored.

➠ The control character (^) may not be used in the text.

❍ Those month names can be used with the commands #YC and #YS.

Example	#DM^ Jan^ Feb^ Mar^ Apr^ May^ Jun^ Jul^ Aug^ Sep^ Oct^ Nov ^ Dec	The names of the months Jan to Dec are stored in the download buffer (each with a closing space).

#DO Deleting one download logo

Definition

Deletion of a single stored logo.

This command deletes a specified logo in the download memory and then releases the corresponding amount of memory.

➡ The command must stand outside of the command sequence #ER to #Q!

Syntax	#DOn	Groups A-G
	#DOn/m	Group H

n = int Reference number of the logo to be deleted (0 to 255).

m = analog to m = A.
= A Deletes the logo on the internal RAM disk.
= C Deletes the logo on CompactFlash card.

Example

```
#G --------------------------------------------------------------------------------------------
#G Logo number 5 is deleted (consultation) and the corresponding
#G memory capacity is made available again.
#G --------------------------------------------------------------------------------------------
#DO5
```

E

#ER Start of label format

Definition Start of a label format (print job) and definition of general information.

➡ Print jobs must be started with the command #ER and closed with #Q!

Syntax	#ERxr/n/b	Group A
	#ERx/n/b	Groups B, TTX 300 online/ Cobra online
	#ERxr/n/b/d	TTX 207 / Da Capo
	#ERxyr/n/d	Group G
	#ERxyr/n/b/d	Group E without TDI
	#ERuxyr/n/b/d	TDI
	#ERxrsftw/n/b/d/TEXT	Group H

u = H Sets the parameter SYSP > LPOS to the value HEAD.
= D Sets the parameter SYSP > LPOS to the value DEPO.

x = N No change label
= Y Change label after series (a change label is somewhat longer than the preceding labels - this can be useful, for example when using a stacker. The longer change label then protrudes out of the label stack and makes it easier to sort the labels).

y = I Label is printed inversely
= (blank) Normal print

➡ Caution limitation: During inverse printing, misprints of update fields (number field, clock, line repeat, Easyline).

r = int No. of printed labels before cut

0: No cut in the entire series.
: same as 0.
1: Each label / label row is cut.
2: Cut after every second label / label row.
x: Cut after every xth label / label row.

s = D (64-xx, ALX 92x, DPM, PEM only) Performs a dot check after the printjob has been finished.

❍ For detailed information on the dot check, see topic section "Info-printouts and parameters", parameter SERVICE FUNCTION > Head dot test.

f = C Thermotransfer printing with ribbon autoeconomy switched *off*
= F Thermotransfer printing with ribbon autoeconomy switched on
= T Thermal printing with printhead lift switched *off*
= U (64-xx, ALX 92x, DPM, PEM only) Thermal printing with printhead lift switched on

t = Pnum (P0.0…P10.0) Delay time in seconds, until a new bit image is generated.

w = After a stop by switching the printer offline, „Please wait…“ is displayed until the delay time (see above) is over. Only then, the printing process can be continued. The demanded delay time can *not* be overridden by quickly switching the printer offline and online again.

A = TTX 600 --TLX 604 – Eagle **B** = TTX 300 Offline - Cobra Offline **D** = TTX 300 Online –TTX 207 - Cobra Online – Da Capo
E = TTX 450/650/674/675/950/1050 – Puma/Puma plus/Lion/Lion plus/Tiger/Tiger XXL -- S45/65/95/105 - TDI/STDI/XXTREME
G = TTX 350 – Ocelot – TTK – Texxtile **H** = 64-xx – Chess x – DPM – PEM – ALX 92x – AP 4.4 – AP 5.4 – AP 5.6 – AP 7.t

= V — A stop by switching the printer offline is executed immediately, without waiting for the delay time to be over. Thus, the demanded delay time can be overridden by quickly switching the printer offline and online again.

n = int — Number of labels in a label row. A label row is a line of labels rectangular to the feed direction. All labels in a row are printed at the same time.

0: One label per row
: One label per row
1: One label per row
2: Two labels per row
x: x labels per row

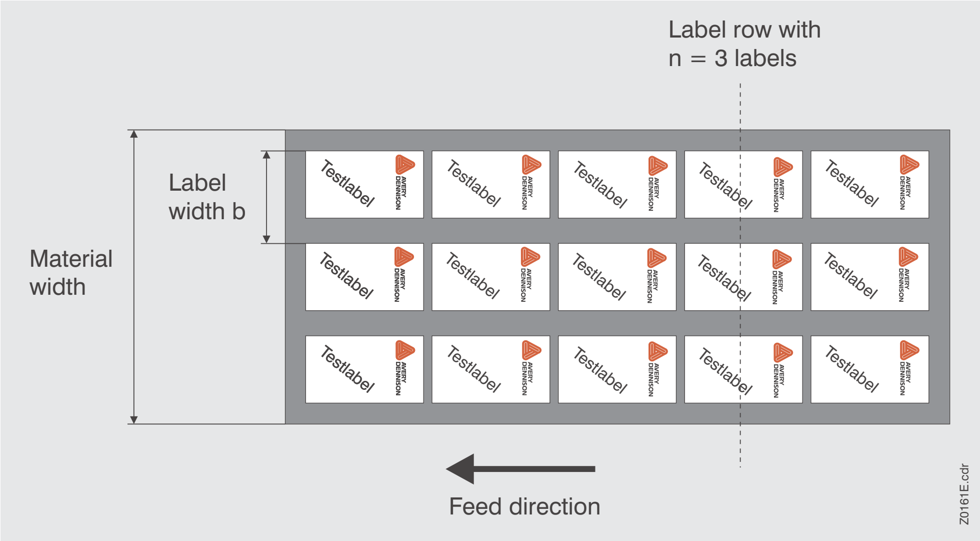


Fig. 1 Illustration of the parameters b and n at 3-track label material (three labels are printed at the same time side by side).

b = num — Width of a label

➠ Parameter b has only to be set for the use of multitrack label materials (n > 1).

➠ The *number of tracks* n multiplied by the *label width* b must not be larger than the *material width*, otherwise the overhanging track will not be printed.

d = num — Width of the double cut (x.xx mm)
Maximum possible double cut = 5.00 mm

➠ If the double cut command is processed without specifying d, the value of the last processed print job or the value set by parameter menu will be used.

TEXT = — Printjob name (alphanumeric text with max. 255 characters string length)

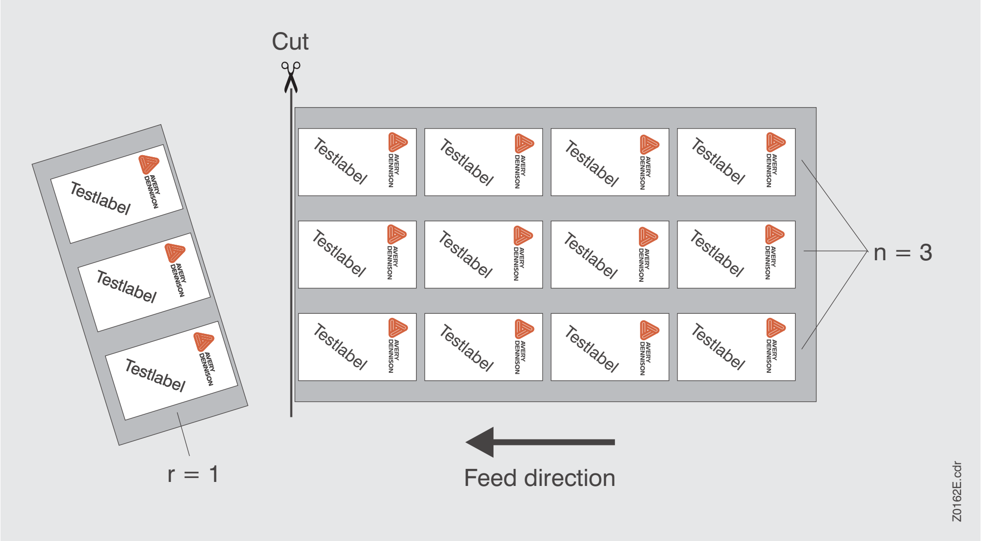


Fig. 2 Example of a setting with n = 3 and r = 1.

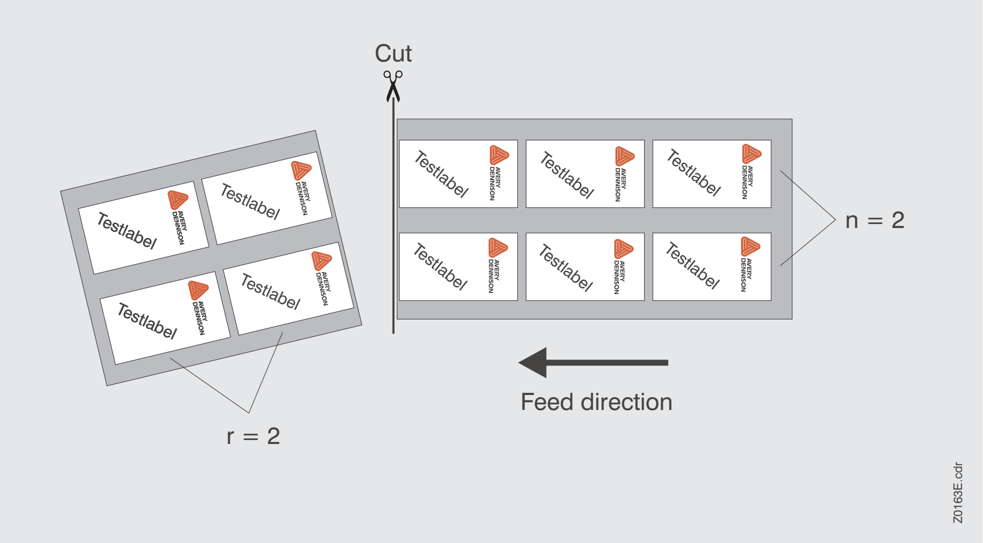


Fig. 3 Example of a setting with n = 2 and r = 2.

	#ERN	Print series without change label
Example 2	#ERY2/3/40	Print with change label at the end of the print series. Only every second label is cut. 3 labels 40 mm wide are printed next to each other.

F

#FC Material feed with cut

Definition

Material is fed through by one label length (only outside of a format) and then cut.

With reel material:	feed by a defined label length
With gapped material:	feed to the next gap

"Gapped" materials are materials with positional markings (punched holes, reflecting markings and register gaps on self-adhesive labels).

➠ The command must stand outside of the command sequence #ER to #Q!

➠ The command does not work in label formats for standalone mode!

Syntax	#FC	Groups A, E, G, H

#FD Field orientation and options

➠ Command must be placed between #ER and #Q!

Syntax	#FD/d/pz#G	Group H (without AP 4.4 / 8MB)

d = int (0 to 3) Rotation direction for fixfonts
0: normal writing direction
1: Text rotated 90 degrees
2: Text rotated 180 degrees
3: Text rotated 270 degrees

d = Dnum (D0.0 to D359.9) rotation direction for scalable fonts

p = L Field position: left justified. The field is built up from the print position to the right.
= M Field position: centered. The field is built up from the print position to both sides.
= R Field position: right justified. The field is built up from the print position to the left.

z = P Normal black printout
= A “White printout”, that is the printing is left blank. Requires a dark background
= E Printout with inverted bitimage (black is left blank; white is printed black)

#FE Material feed with cut(s)

➡ Only with TTK/Texxtile, TTX 350/Ocelot and AP 7.t

Feeds and cuts the labels remaining between printhead and cutter at the end of a print job. The label length defined in the last print job is used.

➡ The command must stand outside of the command sequence #ER to #Q!

➡ The command does not work in label formats for standalone mode!

Syntax	#FE	Group G, H (AP 7.t only)
#FE =	Feed and cut of the remaining labels at the end of a print job.	

#FF Material feed

Definition Material is fed through by one label length (only outside of a format).

With reel material:	feed by a defined label length
With gapped material:	feed to the next gap

"Gapped" materials are materials with positional markings (punched holes, reflecting markings and register gaps on self-adhesive labels).

➡ The command must stand outside of the command sequence #ER to #Q!

➡ The command does not work in label formats for standalone mode!

Syntax	#FF	All groups
#FF =	Feed by one label length or up to the next gap.	

#FO Read in Easy-Plug file

Definition This command is a placeholder of the therin specified file.

If the Easy-Plug interpreter hits a #FO command, it jumps to the specified file and interpretes the therein contained commands. After finishing this file, the rest of the initial command file is interpreted.

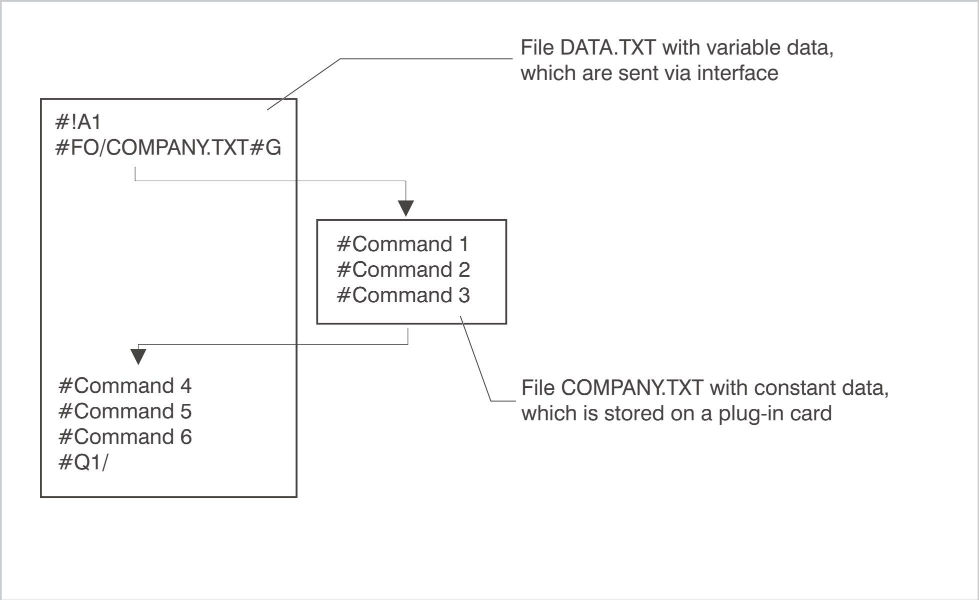

File DATA.TXT with variable data, which are sent via interface
#!A1
#FO/COMPANY.TXT#G
#Command 1
#Command 2
#Command 3
#Command 4
#Command 5
#Command 6
#Q1/
File COMPANY.TXT with constant data, which is stored on a plug-in card

Fig. 4 Inserting commands from a file which is stored on a plug-in card into the initial format file (see also examples 2 and 3).

The advantage of this file separation is the reduced amount of data, which has to be transmitted to the printer. The often more voluminous constant data can be stored in a file on plug-in card or on RAM-disk (only Group H).

Syntax #FO/f#G Groups E, G, H

f = Filename; must be according to the DOS name convention (drive, path, name, extension).

Always write file names in *capital letters*!

The file must already be copied to the printer RAM or the CompactFlash card (see #DF).

Group H only:
The following drives may be selected:
A: Internal RAM
C: CompactFlash card

#G = The command #FO must be closed with #G

Example 1 Command call	#FO/AFTER__8.FOR#G	Command call for groups E, G
	#FO/C:\PFAD\...\VERZ_1\AFTER__8.FOR #G	Call for group H

➠ The file name extension may be choosen any way, it may be *.TXT or *.FOR or any other three-letter-combination.

Example 2 Format file COMPANY.TXT (This file is stored on plug-in card)	#IMN70/50	Label size 50x70 mm
	#ERN0	Start of label format
	#T5#J40#YT104/0///COMPANY #T5#J30#YT104/0///located #T5#J20#YT104/0///in GERMANY	
	#T5#J20#YT104/0D///$00,15#G	Field 00 with 15 characters
	#T5#J02#YB1/0D/10/3///$01,12#G	Field 01 (bar code) with 12 characters

To be able to use the file COMPANY.TXT, the following (example) data must be transmitted from the PC (mainframe) to the printer:

Example 3 File DATA.TXT (file is transmitted to the printer)	#!A1	Activate printer
	#FO/COMPANY.TXT#G	Insert COMPANY.TXT
	#YV00/Denmark	Field 00 with content „Denmark“
	#YV01/999333777001#G	Field 01 with content „999333777001“
	#Q7/	Print amount 7
	#YV00/USA	Field 00 with content „USA“
	#YV01/444197666001#G	Field 01 with content „444197666001“
	#Q5/	Print amount 5

G

#G End of command

Definition

This command is used for closing a preceding command if the following applies:

– the last parameter of the preceding command was of type num, int or TEXT
  and
– the preceding command is the last command of the format.

Additionally, #G can mark a comment line in an Easy Plug file. A comment line, headed by #G may contain nearly every possible character combination – with one exception:

➡ Don´t start a comment line with slash or blank+slash:

#G/
#G /

You may use as many slashes as you want further behind:

#G L/ong live the Queen /////

Syntax	#G	All groups

#G =

Command end for possibly affected commands:

#IM, #ER, #T, #J, #DK, #DC, #DO, #FO and #Y commands.

#G should always be set for closing commands for

– quantity details,
– downloading a logo
  and
– bar codes, for which the characters <20hex are permitted (e.g. Code 128, EAN 128)

Example	#Q100#G	Number of labels = 100 and the numerical value is closed with a blank command.

H

#HR Head resistance adjustment

Definition Sets the head resistance to the required value n (in ohms). This allows the head temperature to be adjusted to the print head.

Syntax	#HRn	Group D Group B (TTX 300/Cobra offline only)

n = int Head resistance (425 to 575 for TTX 300) (500 to 600 for TTX 207)
425 or 500 = lowest head resistance
575 or 600 = highest head resistance

➡ Incorrect head resistance settings reduce the service life of the print head.

#HV Head temperature adjustment

Definition Sets the head voltage and therefore the head temperature to the required value n.

➡ Command must stand outside of #ER and #Q!

Syntax	#HVn	Groups E, G, H

n = int Head temperature (0 to xxx)
0 = lowest head temperature
xxx = highest head temperature (value depends on the printer type, see table below)

Printer	max. HV value
TTX 67x, TDI, TTK, TTX 350, ALX 720	99
64-xx, DPM, PEM, ALX 92x	110
AP 4.4, AP 5.4, AP 7.t	120

⚠ CAUTION!

The printhead temperature (= HV-value) directly affects the life of a printhead. It counts: “The higher the temperature, the printhead is driven with, the lower is its life durance”. This counts even more if HV-values above 100% are driven. Therefore note:

➜ Always choose the lowest possible HV-value necessary to produce an acceptable print result.

I

#IDM Data Matrix Code

Definition

Prints the 2-dimensional "Data Matrix" bar code.

To print the Data Matrix Code, the standard bar code command #YB has to be replaced by the special command #IDM.

➡ Use of the ECC 200 Level (Error checking and correction algorithm), without any possibility to change the level.

Groups E, G:
The appropriate bar code card (Part number A0118) is required for the Data Matrix Code.
❍ Refer to the Plugin Card Manual, topic section "Card Types"

Syntax	#IDM/d/s/vo/a/TEXT#G #IDMn/idgwrc/s/vo/a/TEXT#G	Groups E, G Group H

n = int

0 = ASCII
1 = C40
2 = TEXT
3 = BASE256
4 = reserved
5 = AUTO (Default setting)

INFO: Data Matrix encoding methods

- ASCII
  Encoding of ASCII data, double density numeric data and symbology control characters.
- C40
  Encoding of ASCII data. Packs three alpha numerical data characters into two code words. Usage if data contains more upper case than lower case characters.
- TEXT
  Encoding of ASCII data. Packs three alpha numerical data characters into two code words. Usage if data contains more lower case than upper case characters.
- BASE256
  Encoding of any 8 bit data.

➡ Set SYSTEM PARAMETERS > Character filter to „All characters“, if all characters from 0x00 to 0xff are supposed to be coded!

i = B EAN/UCC mode with data designator put in parantheses. Data has to be sent in parantheses, but the parantheses will not be printed coded. Same handling as EAN 128 bar code.

= X EAN/UCC mode with data designator not put in parantheses. Data has to be sent without parantheses. Same handling as EAN 128 bar code.

d = 0 Normal write direction
= 1 Text rotated by 90 degrees
= 2 Text rotated by 180 degrees
= 3 Text rotated by 270 degrees

g = D Text field consists of one variable data field.

w = W Counter field TEXT is incremented/decremented without a carry over, i.e. only the first unit position of the figure is increased/decreased.

r = Rn n = Number of rows

c = Sn n = Number of columns

As a standard, the size (number of rows and columns) is calculated automatically in dependance of the input data (smallest possible matrix). Use the parameters r and c to set the size.

Row	Column	Row	Column
10	10	64	64
12	12	72	72
14	14	80	80
16	16	88	88
18	18	96	96
20	20	104	104
22	22	120	120
24	24	132	132
26	26	144	144
32	32	8	18
36	36	8	32
40	40	12	26
44	44	12	36
48	48	16	36
52	52	16	48

Tab. 2 Admissible combinations of rows (r) and columns (c).

Example	#IDM/0R12S12/16/0/1/00000A89#G	The data will be coded in a 12x12 matrix.

s = int Number of printer dots for a Data Matrix Block (1 to 60)

v = + Incrementation – offset is added to the text field.
= - Decrementation – offset is subtracted from the text field.

o = int Offset, which is added to (incremented) or subtracted from (decremented) TEXT depending on the prefix.

a = int Number of labels with the same constant number (1 to 255).

TEXT = User data; can be any alphanumeric code with a maximum length of 1024 characters.

Only group H:
The text field may contain an *input field*.

❍ Syntax description: See chapter Input Fields on page 4.

Only group H:
The text field may also be a *variable data field*. Precondition: the D-flag must be set.

❍ Syntax description: See chapter Variable Data Fields on page 3.

#G = The command #IDM must be closed with #G.

INFO: Optional control characters

Not in EAN/UCC mode with BASE256 encoding!

~X is used to represent character values from 0 to 26. Replace the X like in the following examples:

- ~@ = 0
- ~A = 1
- ~B = 2
- ~C = 3

~1 represents the character FNC1. If FNC1 appears in the first position (or in the fifth position of the first symbol of a Structured Append), it will indicate that the data conforms to the UCC/EAN Application Identifier standard format.

~2 is used to represent Structured Append. Structured Append is used to link information from several symbols in a sequence. The ~2 must be followed by 3 additional bytes. The first 4 bits of the first byte identify the position of the particular symbol in the sequence. The last 4 bits identify the total number of symbols in the sequence. The second and third bytes are used as a file identifier and can have a value between 1 and 254 (up to 254*254=64516 identifiers). See Data Matrix Specification for more information about this (ISO 16022).

~3 is only allowed in the first position of the symbol. It indicates that the data contains commands for the barcode reader.

~4 Not allowed.

~5/~6 are only allowed in the first position of the symbol. If ~5 is used the header [)> ASCII*30* ASCII*05* ASCII*29* will be transmitted by the barcode reader before the data in the symbol and the trailer ASCII*30* ASCII*04* will be transmitted after the data. If a ~6 is used, the header [)> ASCII*30* ASCII*05* ASCII*29* will be transmitted by the reader before the data and the trailer ASCII*30* ASCII*04* will be transmitted afterwards.

~7NNNNNN specifies the Extended Channel to be used, where NNNNNN is a value between and 000000 - 999999. For example: ~7000010 means Extended Channel 10. Extended channel is used for using other character sets other than ASCII. See Data Matrix Specification for more information about this (ISO 16022).

~dNNN represents the ASCII character encoded by the 3 digits NNN. For example, ~d065 represents the character 'A'.

#IM Material information

Definition

Definition of the necessary material information and an (optional) material designation.

If the new material designation (parameter c) differs from the previous one, this information is shown on the display to inform the operator that it may be necessary to change the material.

Confirm the message with the ENTER button, return to on-line mode by pressing the ENTER button again.

➡ If a print job does not contain any #IM-command, the material information defined in the previous print job will be used.

➡ Command must stand outside of #ER and #Q!

Syntax		
	#IMxb/l/c	Group B, TTX 300 online/Cobra online
	#IMxyb/l/c	Groups A, E, G, TTX 207/Da Capo
	#IMxyb/l/c/tg/d/ef/h	Group H

x = N For reel material without gaps

= S For material with gap

= X For single material at infeed Group E

y = B For batch mode:
Total surface of the label is printable. In the case of printers in Group A, the last label of a print job is only cut (if programmed) when the next print job is carried out; high output volume.

= E For normal 1:1 mode:
The initial zones of the label are not printable. All labels in a print job are cut (if programmed); high output volume.

= R For real 1:1 mode: Groups E, H, TTX 207/Da Capo
Total surface of labels is printable. All labels are issued (if programmed); reduced output volume.

b = num Material width:
000.00 mm to max. width, according to print head.

l = num Label length:
000.00 mm to max. length, depending on the configuration of the printer.

❍ Find the maximum label length of the concerned printer on Info-Printout STA0 or STA1 (groups A to G) or “Memory Status” (group H), respectively.

c = TEXT Material designation with alphanumerical characters.
– Groups A, B, D, E, G: max. 4 characters
– Group H: max. 16 characters

➡ The character # may not be used.

. t = int Tag type
0 = autodetect (default)
≠ 0 (compare Tab. 3 Supported transponder types.)

g = C Check valid tag (for Alien tags only, read EPC and compare with Alien pattern that marks valid tags “A5A5”) - On alien tags there is an EPC written by Alien. If this EPC begins with Hex A5A5 it is a "good" one (good performance/good quality). If it does not start with A5A5 it means it is NOT a "good" one. But anyway you may be able to write and read these tags.For the purpose of eliminating these tags we check for A5A5 and if there is no A5A5 we invalidate these tags (cross out).

= D Detect whether there is a tag to operate on. Applies to tags without UID/TID. Detected identification is treated internally as UID/TID. Do not use this option if you would like to operate on zero initialized EPC tags (they cannot be detected). This is default value.

= N none

d = 0 No RFID-operations.

= int (d > 0) Distance in print direction (x) from label edge to optimum of transponder antenna.

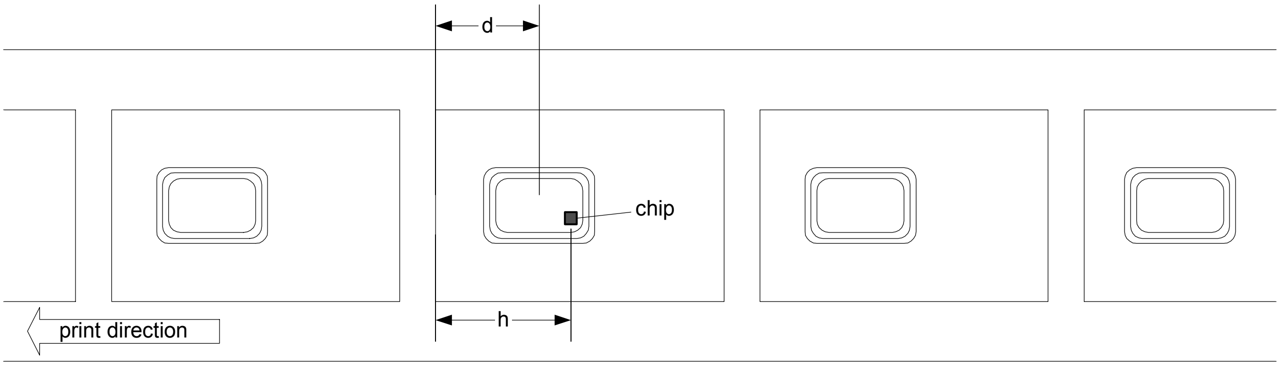
➡ It is recommended to set parameter y to R when parameter d is used. This parameter is mandatory for any RFID processing (even if distance is zero).

e = int Number of retries with different tags in case of detect errors. Default defined by parameter “Max tags to stop”. Range from 0 to 10.

f = V verify option (default: don’t’ verify) - Verify written data with an extra read after write and compare it. If written data and read data is different an error is reported

h = int Distance to middle of chip from label edge, to activate head lift for chip protection. With h option we define an explicit range of head lifting.

➡ With this setting, print data in the chip area is not printed.



»

Examples	#IMS50/100/TYP1	Gapped material, 50 mm wide, 100 mm long, TYPE 1
	#IMN20/50/	Reel material, 20 mm wide, 50 mm long
	#IMSR98/165///70	Gapped material, 98 mm wide, 165 mm long, with RFID transponder; 70 mm distance from label edge to middle of transponder antenna, autodetect transponder type
	#IMSR98/165//6/70/	Gapped material, 98 mm wide, 165 mm long, with RFID transponder; 70 mm distance from label edge to middle of transponder antenna, transponder type “EPC class 1”

INFO: Supported RFID tag types

Manufacturer	Tag type	ID
Matrics	EPC class 0	5
Alien	EPC class 1	6
Philips	U•CODE EPC 1.19	10
Philips	U•CODE HSL	11
Impinj	Zuma / EPC class 0+	12
Various	EPC class 1 Gen2	15

Tab. 3 Supported transponder types.

J

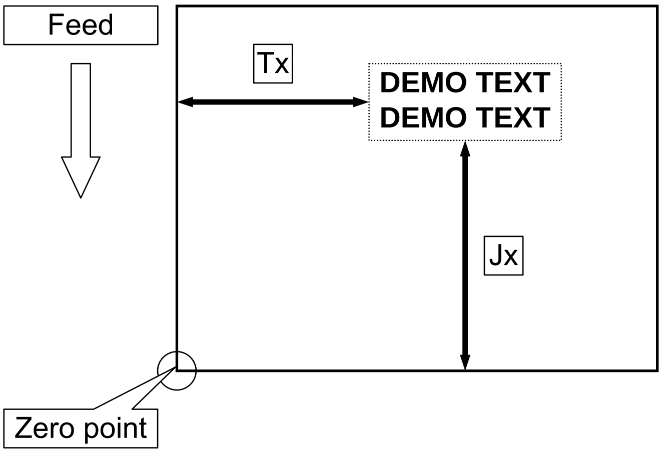
#J Vertical print position

Definition Determines the vertical print position with absolute value in mm.

Zero position is always the bottom, left-hand corner of the label. Exception: command #R is active.

➡ Command must stand between #ER and #Q!

Syntax	#Jx	All Groups
x = num	Vertical print position in mm in relation to the bottom left-hand corner of the label (000.00 mm)	



➡ Text outside of the defined label area is not printed.

➡ Reference line of the text is the base line. A base line, which exceeds the defined label area, causes an error message, even if there are no descenders in the text.

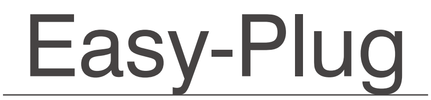


Fig. 5 Base line of a text. Even if the text has no descenders (as the y and the g in the example text), the base line lies a certain distance below the text.

Example	#J5	Text begins 5 mm from the bottom.
	#J20	Text begins 20 mm from the bottom.

L

#LC Initialising logo card

Definition The logo card (SRAM card) in the slot is erased and initialised. A reset is then carried out automatically.

➡ All of the data on the logo card is erased.

New logos can be transmitted to the card with the command #DK.

Syntax	#LC	Groups D (without TTX 207), E, G

Example	#LC	Logo card is initialised and a reset is carried out.
	#!A1	Printer switched to on-line mode, incoming commands are processed.
	...	
	#Q.../	

M

#M Magnification factor

Definition

This command enables characters and logos to be magnified in the X/Y axis by the defined factor (1 to 8).

All characters which follow are magnified by the defined factor until #M1/1has been sent, the printer reset or a new label series started.

➡ Command must stand between #ER and #Q!

Syntax	#Mx/y	All groups

x = int

Magnification of the font width (run direction)

Groups A, B, D, E, G	1 to 8
Group H	1 to 16

y = int

Magnification of the font height

Groups A, B, D, E, G	1 to 8
Group H	1 to 16

Example		
	#M1/1	Characters are printed normally.
	#M2/4	Characters are magnified by a factor of 2 in the X axis and by a factor of 4 in the Y axis.
	#M4/1	Characters are magnified by a factor of 4 in the X axis and are printed normally in the Y axis.

#ME Eject material

Definition

- Cuts the label material off.
- The printer ejects the label material by moving it backwards.

➡ The command #MOF may only be sent outside of a format #ER to #Q/.

➡ Use this command only for printers equipped with a cutter!

➡ The command does not work in label formats for standalone mode!

Syntax	#ME	Group H

#MOF Material end detection Off

Definition

Deactivation of the sensor for detecting the material end.

➡ The command #MOF may only be sent outside of a format #ER to #Q/.

Syntax	#MOF	Group D

#MOF =

Deactivation of the material end detection.

Default setting for the material end detection is "activated". The value is stored in the NOVRAM.

A = TTX 600 --TLX 604 – Eagle **B** = TTX 300 Offline - Cobra Offline **D** = TTX 300 Online –TTX 207 - Cobra Online – Da Capo
E = TTX 450/650/674/675/950/1050 – Puma/Puma plus/Lion/Lion plus/Tiger/Tiger XXL -- S45/65/95/105 - TDI/STDI/XXTREME
G = TTX 350 – Ocelot – TTK – Texxtile **H** = 64-xx – Chess x – DPM – PEM – ALX 92x – AP 4.4 – AP 5.4 – AP 5.6 – AP 7.t

#MON Material end detection On

Definition Activation of the sensor for detecting the material end.

➡ If the sensor is deactivated, the printing process must be stopped manually at the end of a roll, otherwise the print roller could be damaged.

The command #MON may only be sent outside of a format #ER to #Q/.

Syntax	#MON	Group D

#MON = Activation of the material end detection.

Default setting for the material end detection is "activated". The value is stored in the NOVRAM.

Example		
	#MON #ER ... #YB0/0/10////12345678	The material end sensor is activated.

#MXC Bar code MaxiCode

Definition Prints the 2-dimensional "MaxiCode" bar code.

(Only groups E, G)
The appropriate bar code card is required for the MaxiCode.
❍ Refer to the Plugin Card Manual, topic section "Card Types"

The Easy Plug command #YB for bar codes is replaced by the special command #MXC for the Maxicode.

Syntax	#MXCz/d/x/y/vo/a/TEXT#G #MXCz/dw/x/y/vo/a/TEXT#G	Groups E, G Group H

z = int MaxiCode modes 2, 3, 4 and 6

d = 0 Normal write direction
= 1 Text rotated by 90 degrees
= 2 Text rotated by 180 degrees
= 3 Text rotated by 270 degrees

w = W Counter field TEXT is incremented/decremented without a carry over, i.e. only the first unit position of the figure is increased/decreased.

x = int Barcode number for code-splitting onto more than one barcodes (1 to 8; default: 1)

y = int Amount of barcodes in case of code-splitting onto more than one barcodes (1 to 8; default: 1)

v = + Incrementation – offset is added to the text field.
= - Decrementation – offset is subtracted from the text field.

o = int Offset, which is added to (incremented) or subtracted from (decremented) TEXT depending on the prefix.

a = int Number of labels with the same constant number (1 to 255).

TEXT = Depending on the mode (z), the following information:

z	TEXT (Syntax)	Char.	Data
2	ppppppppp ccc sss MSG	9	Numerical postcode
		3	ISO national code
		3	Service class
		84	Code words *)
3	pppppp ccc sss MSG	6	Alphanumerical postcode
		3	ISO national code
		3	Service class
		84	Code words *)
4	MSG	93	Any alphanumerical code
6	MSG	93	Any alphanumerical code

Tab. 4 Structure and length of the data string "TEXT" depend on which mode "z" is selected.

*) *Depending on the selection of alphanumerical characters (0x00 – 0xff) results a different amount of user data, because more or less switches between subsets are necessary.*

➡ In Modes 2 and 3 a space must be left respectively between the postcode, national code, service class and user data. The number of characters in the postcode, national code and service class must be observed exactly.

➡ (Only group H) The text field may contain an *input field*.

❍ Syntax description: See chapter Input Fields 🗋 on page 4.

#G = The command #MXC must be closed with #G.

N

#N National character set

National Character Set

Definition Changes the fonts between different national character sets.

➡ Command must stand outside of #ER and #Q!

Syntax #Nn All Groups

n =	0	USA
=	1	UK
=	2	France
=	3	Germany
=	4	Italy
=	5	Sweden
=	6	Spain
=	7	Norway
=	8	Special
=	9	IBM-similar
=	10	ANSI codepage 1252 Latin 1 (equals ISO 8859-1 Latin 1)
=	11	(Group H) ANSI codepage 1250 Central Europe
=	12	(Group H) ISO 8859-2 Latin 2
=	13	(Group H) UTF-8

Example
Printing of russian text with UTF- 8 coding

```
#G ------------------------------------------------------------------------------------------
#G Prerequisite: The cyrillic font is stored on a memory card in folder
#G \fonts. Filename: „font900.xxx“.
#G ------------------------------------------------------------------------------------------
#!A1
#IM200/100
#N13
#ER
#SS900/OV/32/#G
#T01.0#J010.0
#VW/L/“Указания по тезнике безопасности при эксплуатации машины
для печатания этикеток“
#Q1/
```

O

#OLVI Initialize Online Verifier

Definition Send an initialization string to the online verifier (OLV). Only valid for the OLV type RJS SV100.

➡ Command must stand *outside of* #ER and #Q!

Syntax	#OLVIn/INITTEXT	Group H

n =	0	INITTEXT will only then be sent to the OLV, if the content of the string has changed.
=	1	INITTEXT is always sent to the OLV.
INITTEXT =		SV100 commands, which are sent to the OLV before a print job is started.

❍ For admissible initialization commands refer to the SV100 manual.

#OLVD Define OLV limits

Definition Definition of limits for the separate readability criteria. Is only valid for RJS SV100 OLV.

➡ For every bar code may be defined a separate limit.

➡ Command must stand *between* #ER and #Q!

Syntax	#OLVD/a/b/c/d/e/f/g/h/i/j/k/u	Group H

a =	P	Decodable (Pass)
=	F	Not decodable (Fail)
b =	0...100	Decodability
c =	0...100	Modulation
d =	0...100	Defects
e =	0…100	Edge contrast
f =	0...100	Rmin/Rmax
g =	0...100	Symbol contrast
h =	0…100	PCS
i =	0...100	R (white)
j =	0...100	R (black)
k =	0...99	Ratio
u =	0...40	ANSI symbol grade

➡ The measured value must in all cases be above or equal the set limit.

❍ For detailed information please read the SV100 manual.

Example	#OLVD//45//20////////	Only „Decodability“ and „Defects“ is analysed.

P

#PA Offset print start

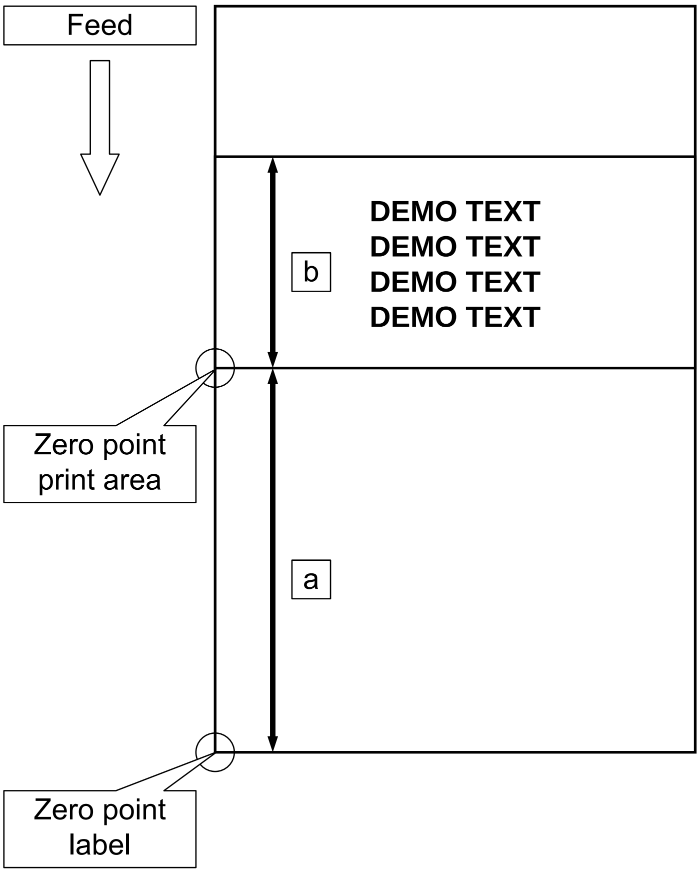
Definition Offsets the beginning of the print range by the given value. Select autoeconomy mode to lift the print head while the material is being fed to the print range.

➡ The command must be sent before the command #IM!

Syntax	#PAa/b	Groups E, G, H

a = num Offset in mm from the physical start of the label until the print start

b = num Length of the range to be printed



Example		
	#PA400/100	Length of the print area 100 mm, offset 400 mm from where the label starts
	#IMSB/100/600	Label width 100 mm, total length 600 mm

#PC Setting Parameter Values

Setting of parameters in the printer menu to certain values. The function is helpful, if…

- settings must be transferred from one printer to another.
- several printers are supposed to have the same settings.
- the settings of a printer are supposed to be recovered, e.g. after the CPU board was replaced.

It is adviseable to read out the parameter settings of the concerned device at first. This can be done one of the following ways:

- With the Easy-Plug command #!PG.
- By aid of parameter SPECIAL FUNCTION > Parameter to CF .

➨ Many of the parameters need a system reset, before the changed setting becomes effective. This can be triggered at the printers operation panel or by an Easy-Plug command.

Syntax #PC<ParameterID>/<Value>#G Group H

<ParameterID> *Parameter Ident number (ID)*. Each parameter in the menu has its own Parameter-ID.

➨ Finding out a Parameter-ID: Call SPECIAL FUNCTION > Parameter to CF. The so generated text file contains the assignment Parameter-ID/Parameter.

999999 = Parameter-ID for special functions; must be applied in combination with one of the below listed values (Tab. 5).

/ Separation sign

<Value> *Parameter Value*. The admissible values depend on the type of parameter. Assigned values can be integer numbers, floating point numbers or strings.

➨ The parameter values for the parameters IP address, Netmask and Gateway address can be specified in the format xxx.xxx.xxx.xxx or as plain integer.

❍ Further information can be found in the command description of #!PG.

Value	Function
-1	Reset
1	If the parameter SPECIAL FUNCTION > Factory settings is called, the current parameter settings are used (equals the parameter setting SPECIAL FUNCTION > Default values = User defined)
2	If the parameter SPECIAL FUNCTION > Factory settings is called, the standard factory settings are used (equals the parameter setting SPECIAL FUNCTION > Default values = Standard)

Tab. 5 Admissible values for special functions – must be applied together with Parameter-ID „999999“.

Example		
	#PC1508/viper#G	Sets the FTP Server password to „viper“ (INTERF. PARAM. > ETHERNET PARAM. > FTP password = viper)
	#PC999999/-1#G	The printer will be started newly (reset)

#PDF Bar code PDF 417

Definition

The bar code PDF 417 is a two-dimensional bar code with high character density, a complete ASCII character set, with a high level of security and an excellent ability to compensate for errors.

(Only groups E, G)
The appropriate bar code card is required for the bar code PDF 417.
❍ Refer to the Plugin Card Manual, topic section "Card Types"

The Easy Plug command #YB for bar codes is replaced by the special command #PDF for the PDF 417 bar code 417.

➨ The command must be placed between #ER and #Q!

Syntax		
	#PDFn/td/s/l/TEXT	Group E, G
	#PDFn/td/s/l/z/w/h/TEXT	Groups H

n =		Compression type
=	0	EXC mode (Extended alphanumeric Compaction Mode)
=	1	Binary ASCII Plus Mode
t =	T	Unidirectional reading
=	(blank)	Bi-directional reading
d =	0	Normal write direction
=	1	Text rotated by 90 degrees
=	2	Text rotated by 180 degrees
=	3	Text rotated by 270 degrees
s =	int	Security Level (0 to 8)
l =	0	Bar code width is set automatically
=	int	Bar code width in code words (number of columns) (1 to 30): The width should be so selected that a height of 90 lines is not exceeded with regards to the area of the bar code defined in the user data. A maximum of 928 code words are permitted for one bar code. Of these the following are preassigned: – 1 word for the length input of the code – x words for the check total (Security Level) = 2 (1 + Security Level)
z =	0	Number of lines is set automatically
=	int	Number of lines (3 to 90)
w =	int	Bar code width (1 to 16)
h =	num	Height of a PDF bar code row in millimetres (1 to 100)
TEXT =		User data: Permitted characters depending on the compression type. Max. string length: 1024 characters.

➨ (Only group H) The text field may contain an *input field*.
❍ Syntax description: See chapter Input Fields on page 4.

#G = The #PDF command must be closed with #G.

#PIN Pin on activation/deactivation

Definition Command #PIN activates and deactivates the pin module. The printer can be used as a conventional printer if the pin module is deactivated.

➠ The command must be sent before the command #IM!

Syntax	#PINn	Group D (only Pin On)

n = int

= 0 Pin module deactivated
= 1 Pin module activated

#PO Gap offset

Definition This command is for determining the beginning of the label when irregularly formed labels are being printed.

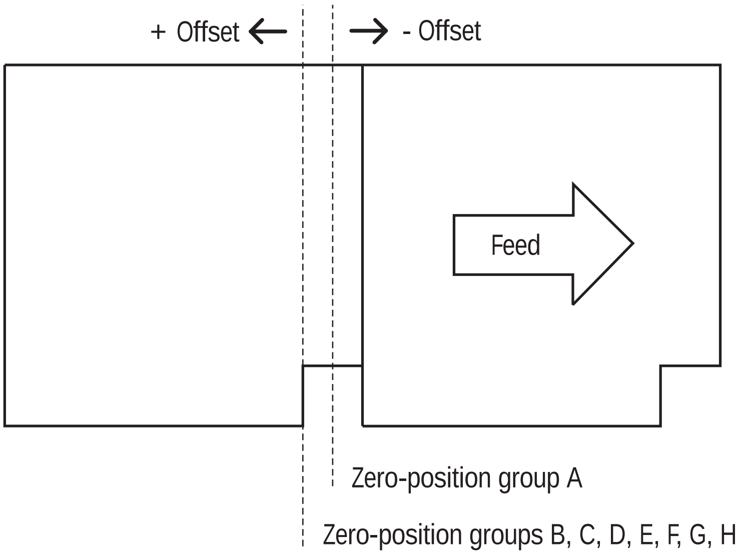
➠ Command must stand outside of #ER and #Q!

Syntax	#POva #POva/s	Groups A, D, E, G, H Group B

v = + Positive offset: label begins before the end of the gap
= - Negative offset: label begins after the end of the gap

a = num Offset in mm:
Observe the limitations of the printer concerned, see User Manual.

s = int Gap length:
Only necessary if labels in bone form are being processed which have the following parameters: gap length >3 mm (reel material) or 15 mm (gapped material).



Example	#PO-20	Beginning of the label 20 mm before the end of the gap
	#PO+5	Beginning of the label 5 mm after the end of the gap
	#PO	Label begins at the end of the gap

#PR Print speed

Definition For setting the print speed and feed speed.

➠ The command must stand outside of the command sequence #ER to #Q!

Syntax #PRx/y/ Groups D, E, G, H

x = int Print speed; The setting range depends on the printer type; setting interval: 1 inch/s

y = int Feed speed; The setting range depends on the printer type; setting interval: 1 inch/s

➠ With DPM/ ALX 92x, decimal places may be programmed. The setting interval then is 0.2 Inch (e.g. #PR 8.6/8.6).

Example	#PR	Both speeds = 8 inch/s (Default setting)
	#PR4/	Print speed = 4 inch/s, Feed speed = Previously set value or, if no value has been set, default setting.
	#PR/6/	Print speed = Previously set value or, if no value has been set, default setting. Feed speed = 6 inch/s
	#PR8/10/	Print speed = 8 inch/s Feed speed = 10 inch/s

#PS Programme scanner

Definition The scanner is programmed with the special command #PS. A maximum of 8 codes (n = 8) can be programmed on one label. This number must correspond to at least the number of activated codes (a, b, c etc.).

➠ The command must stand outside of the command sequence #ER to #Q!

➠ The sequences STX (02h), ETX (03h) and EOT (04h) may not be used in the code.

Syntax #PSa/w/b/x/c/y/d/z/n Groups E

a, b, c, d	=	int	Easy Plug bar code number (0 to 4, 7 to 9, 13, 16)
w, x, y, z	=	int	Number of characters in the bar code (0 = variable bar code length)
n	=	int	Number of codes on the label

➠

The following bar codes are supported:

Barcode	Easy Plug number	Number for scanner
no code		
EAN-8	0	0
EAN-13	1	1
UPCA	2	2
Code 93	3	3
2/5I	4	4
Code 39 (2:1)	7	7
Codabar	8	8
UPCE	9	9
Code 128	13	13
EAN 128	15	13
Code 39 (3:1)	16	7
Code 128 (UPS)	18	13
Code 39 (1,5:1)	19	7
Code 2/5 i (1:3)	20	4
Code 128 A	24	13
Code 128 B	25	13
Code 128 C	26	13
Code 128 Pharmacy	27	13

If several codes of the same type (of the same or any length) are to be printed on one label, it is only necessary to activate the code once and to enter the appropriate number of codes with n.

Example	#PS7/10///////2	Label with 2 bar codes of type Code 39 with 10 positions

➡ If several codes of the same type are activated on one label, they must contain different information.

Q

#Q Print quantity

Definition

Format end and definition of the number of labels to be printed. The format is checked and stored in the memory.

Before the format is stored, it is checked and WAIT is displayed on the operating panel (only in Groups B and C.) Labels are only printed with this new format after it has been taken over correctly.

The format cannot be altered or extended once it has been stored in the memory. Format commands are not evaluated until the memory is empty again. The memory is empty when the label series has been printed or broken off with #CF.

➠ This must be stopped before the printer is switched off (#SP or ON/OFF button for off-line operation) to ensure that the label series is definitely terminated.

➠ Print jobs must be started with the command #ER and closed with #Q!

Syntax	#Qan/	Group H
	#Qn/	Groups A, B, D, E, G

a = A		Standalone mode: The label amount is queried.
= X		Standalone mode: The label amount is *not* queried, but amount “n” is printed.
n =		same as n = 0
= 0		Memory is emptied. Nothing is printed.
	➠	Standalone mode: label amount = endless.
= int		Number of labels to be printed (1 to 65000) -- Groups A, B, D, E, G: 1 to 65000 -- Group H: 1 to 2.2 billion
= *		Star (*) for undefined print quantity (label amount = endless).
	➠	The label number is always rounded off to a multiple of the no. of labels in one utility unit. Example: utility unit with 5 Labels, n = 8, 10 Labels are printed.
/ =		The command #Q must be closed with a slash (/) or #G.

Example	#Q100/	100 labels of the previous format are printed.

R

#R X/Y Offset data blocks

X/Y Offset of all subsequent data blocks

Definition

Logically or optically-related printouts (data blocks) can be shifted as a whole.

The reference point #Tx or #Jx (see #T and #J)for all subsequent positionings is shifted from the zero point by the entered x/y value.

Zero point is the bottom, left-hand corner of the label (see illustration), provided that this has not been shifted by the #PO command.

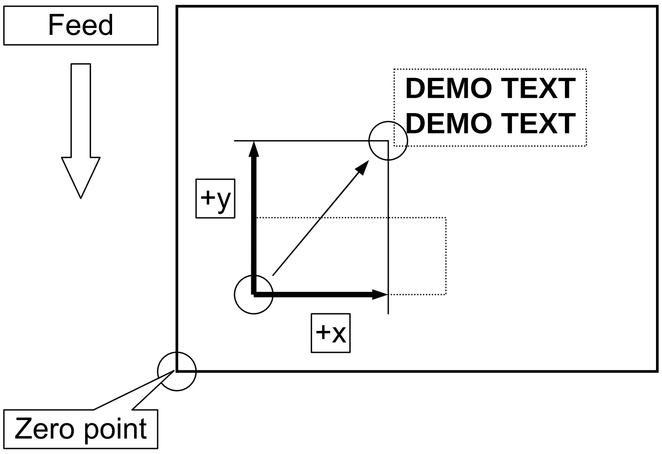
➡ Command must stand between #ER and #Q!

Syntax	#Rvx/vy	Groups B, D, E, G, H

v = + Offset in positive x / y direction
= - Offset in negative x / y direction

x = num Horizontal offset of the print position

y = num Vertical offset of the print position



Example	#R5/5	Block is shifted to the right and upwards by resp. 5 mm.
	#R-5/-5	Block is shifted to the left and downwards by resp. 5 mm.

#RC Coloured ribbon transport

Definition

On printers with colour option (multi-coloured printing), the appropriate colour field is transported on the ribbon under the print head using the command #RC. A maximum of 4 colour fields are possible. The layout can differ. One of the fields (0) assumes the task of co-ordinating colour and impression.

➡ Command must stand between #ER and #Q!

Note

The command #RC defines the colour in which the following elements are to be printed. This colour remains active until a new command #RC activates another colour or until a new label format is defined with #ER.

Syntax #RCn/ Group E

n = int Colour of the selected field.

4-coloured ribbon No. of colours a = 4 (see #CP)

n = 0	black
= 1	yellow
= 2	magenta
= 3	cyan

Other coloured ribbon No. of colours a is not equal to 4 (see #CP)

n = 0	1st colour after the double bar
= 1	2nd colour after the double bar
= 2	3rd colour after the double bar
	etc.

Example

```
#G ------------------------------------------------------------------------------------------
#G This example prints the word GERMANY and colour fields in
#G 8 colours (4-coloured ribbon).
#G ------------------------------------------------------------------------------------------
#!A1
#IMS48/58
#ERN
#RC0/
#J2#T10#M1/1#YT106/0///GERMANY
#J2#T30#YL0/0/5/15
#RC1/
#J8#T10#M1/1#YT106/0///GERMANY
#J26#T10#M1/1#YT106/0///GERMANY
#J32#T10#M1/1#YT106/0///GERMANY
#J38#T10#M1/1#YT106/0///GERMANY
#J20#T30#YL0/0/5/10
#RC2/
#J14#T10#M1/1#YT106/0///GERMANY
#J26#T10#M1/1#YT106/0///GERMANY
#J32#T10#M1/1#YT106/0///GERMANY
#J44#T10#M1/1#YT106/0///GERMANY
```

```
#J15#T30#YL0/0/5/10
#RC3/
#J20#T10#M1/1#YT106/0///GERMANY
#J26#T10#M1/1#YT106/0///GERMANY
#J38#T10#M1/1#YT106/0///GERMANY
#J44#T10#M1/1#YT106/0///GERMANY
#J28#T30#YL0/0/5/10
#Q1/
```

#RFC Special RFID commands

Definition Sends a command sequence to the RFID reader module to trigger special operations which are not mapped / part of the regular read/write/lock operations.

➠ The commands are specific for a RFID reader module.
➠ Only for printers with installed and activated RFID option.
➠ Command must be placed between #ER and #Q!

Syntax #RFC<cmd>#G Group H

<cmd>	=	int	Command code.

---------------- HF-Technology / FEIG reader ---------

(*1) EAS features (only supported by „NXP I-Code“ chips)
16: Set EAS bit
17: Reset/clear EAS bit
18: Lock EAS bit permanently

#RFH Request data - send to host

Definition This command requests data of a RFID variable that has been read and assigned with #RFR command. That data is transmitted as response over selected EASYPLUG interface (same interface that is used for this request).

➠ Only for printers with installed and activated RFID option.
➠ Command must be placed between #ER and #Q!

Syntax #RFHi/m/n/HEAD Group H

i	=	int	number of variable that has been created by read data command #RFR
m	=	int	number of bytes in response, filled if necessary with white space (0x20)
n	=	int	Number of bytes in HEAD
HEAD			characters that precede response

➠➠

Example		Data on transponder (block size 4 assumed e.g. I-Code) shown in hex notation: MSB ... LSB block address 3: 41 42 43 44 block address 4: 30 31 32 33
	#RFR1/0/I/L/2/3	2 blocks from transponder starting at block 3 will be read and put into variable with id number 1. Text in variable 1: DCBA3210
	#RFH1/14/6/BLOCK:	Request var 1 Response: BLOCK:DCBA3210
	#RFR3/2/B #RFH3/20/4/EPC=	EPC in variable 3 Request var 3 Response: EPC=A5A5800F35609854
	#RFR3/1/B #RFH3/20/4/UID=	UID in variable 3 Request var 3 Response: UID=E005000000000C5B

#RFL Lock/unlock memory areas

Definition

Enable or disable the write protection (simple lock) of various memory areas of EPC Gen 2 tags. Before changing the write protection status for the first time, the access password must once have been written into the access password memory of the tag by means of a write command (#RFW or #SRF + #VW/T). For any attempt to change to protection status the same password must be provided as part of the #RFL command.

➡ The #RFL command itself doesn't write the access password into to tag!

The #RFL command doesn't support the permalock option specified in the EPC Gen 2 standard. It implements only the write protection which can be reversed any time by issuing an unlock command with the right access password.

➡ Only for printers with installed and activated RFID option.
➡ Command must be placed between #ER and #Q!

Syntax #RFLa/b/c/TEXT#G Group H

	Operation:
a = 1	Lock (write protect) memory area
= 0	Unlock (unprotected) memory area
	Memory area selection:
b = 0	User memory
= 2	EPC
= 3	Kill password
= 4	Access password

Encoding of input data:

c = B Hex ascii encoding: for the access password 8 characters must be provided
= I Raw data (binary data): for the access password 4 chars must be provided

TEXT AccessPassword
EPC Gen 2 specifies the access password with a size of 32 bit (4 bytes). Depending on the data encoding (option c), 4 or 8 characters must be provided.

#RFR Read data

Definition This command reads data of a RFID transponder and assigns it to a RFID variable. Content of variable can be requested with #!RF command.

➡ Only for printers with installed and activated RFID option.
➡ Command must be placed between #ER and #Q!

Syntax #RFR i/t/ab/s/n Group H

i = int number of variable that is created for read data

t = int Data type
0 = BLOCK [1] [2]
1 = UID/TID [3]
2 = EPC
3 = KILL Password
4 = ACCESS Password
5 = MEM BANK (only read operations)
6 = AFI
7 = MEMBLOCK [1] [2]
8 = DSFID

1) „s“ required
2) „n“ required
3) “n” optional – for limiting size of data returned for read operations.

a = I ASCII data assumed on transponder (default)
Each byte is coded as ASCII character.

= B Binary data assumed:
Each byte is coded as hexadecimal number in ASCII presentation.

b = L Least significant byte of block first (default) :
Bytes of each block on transponder are read beginning with the LSB first.

= M Most significant byte of block first:
Bytes of each block on transponder are read beginning with the MSB first.

s = int If Data type =
-- BLOCK: start block address
-- MEM BANK: address/name of memory bank; (default: 0)
-- MEMBLOCK: combination of memory bank identifier and address offset in the memory bank. Schema for address calculation:
s = <EPC MemoryBankID> * 1000 + <BlockNumber>

EPC MemoryBankID:
0: reserved
1: EPC
2: TID
3: USER
Default setting: 0. Other values are ignored.
-- Other data type: parameter is ignored

n = int If…
-- Data type = BLOCK, TID, MEM BANK or MEMBLOCK: number of blocks to be read (default: 0)
-- Other data type: parameter is ignored

➡ For reading a TID field (t=1), the size parameter can be used to limit the size of the returned data. Giving no size information (n=0) will return the whole content of the TID memory bank, which may contain more information than expected (depending of RFID transponder/chip). Typical values for n are:
2: Read 32Bit Vendor-ID/Chip-ID
4: Read 32Bit Vendor-ID/Chip-ID + 32Bit Unique Serial Number

Example		Data on transponder (block size 4 assumed e.g. I-Code) shown in hex notation: MSB ... LSB block address 3: 41 42 43 44 block address 4: 30 31 32 33
	#RFR1/0/I/L/2/3	2 blocks from transponder starting at block 3 will be read and put into variable with id number 1. Text in variable 1: DCBA3210
	#RFR1//B/3/2	Now binary data Text in variable 1: 4443424133323130
	#RFR1//M/3/2	Now Ascii data reversed Text in variable 1: ABCD0123
	#RFR3/2/B #RFR3/1/B	EPC in variable 3 UID in variable 3

#RFW Write typed data to RF tag

Definition This command writes data of well-defined type onto a RFID label.

➡ Only for printers with installed and activated RFID option.
➡ Command must be placed between #ER and #Q!

Syntax #RFWt/abfs/r /a/n/DATA#G Group H

t = int data type
0 = BLOCK/USER DATA
1 = UID/TID
2 = EPC
3 = KILL PASSWORD
4 = ACCESS PASSWORD

a = I data assumed in DATA as ASCII(default)
Each byte in DATA is written to chip in identical form. There is no interpretation. This is especially designed for printable byte codes as ASCII characters.

= B data assumed in DATA in hex representation:
Two consecutive bytes in DATA are interpreted as hexadecimal description for a byte code. This byte code is written on chip. This is especially designed for NON printable byte codes.

b = L Least significant byte of block first (default) :
Bytes of DATA are written on transponder beginning with the LSB first.

= M Most significant byte of block first:
Bytes of DATA are written on transponder beginning with the MSB first.

f = P locking option (write protect), default: don't' lock (NOT YET IMPLEMENTED)

s = int Address of start block. Default setting: 0 (= Block no.)

r = int Number of read/write retries. Default defined by parameter “Nr of CMD retries”.

n = int Number of characters in DATA

DATA data to be written

➡ (Data type BLOCK only) n must be a multiple of the block size, otherwise padding with 0 to a multiple of the block size.

Example	#RFW2/B///24/01203D2A916E8 B8719BAE03C	12 bytes as an EPC 96 in hex representation

#RM Measurement

Definition This command initiates a test of a certain type and stores results in a defined destination.This command overides any other RFID command. Do not use any other RFID command within the same format.

➠ Command must be placed between #ER and #Q!

Syntax	#RMt/d	Group H

t = int type of test
0 = Measure tag read detects per interval to identify optimized antenna position (= Hot Spot Test)(default)

d = int destination
0 = print histogram on label (default)
1 = save as file to Compact flash card (not implemented yet)
2 = send to EASYPLUG interface (not implemented yet)

Example	#!A1 #IMSR99.0/152.0//15/0/1/0 #ERN// #RM0,0 #Q1#G	Activate Material definition - use exact label size - for EPC tags use option N (No detect) - use antenna offset 0 to start at beginning of label - chip protection switched off Start label Measure number of read detects per interval and print result as bar on label Quantity 1

#RSS GS1 DataBar & CC

GS1 DataBar (formerly RSS) & Composite Component (CC)

Definition Printing of a bar code field with a GS1 DataBar or a composite symbology bar code.

➠ The command must be terminated with #G!

Syntax	#RSSzx/dw/s/vo/a/TEXT#G	Group H

z = int Bar code number

❍ See topic section „Info-printouts & Parameters“, paragraph „Font status“, Bar code library“.

➠ *GS1 DataBar Expanded Stacked* is applied automatically, if the following conditions are met:
-- Selected barcode type: GS1 DataBar Expanded (z = 6)
-- Width (x) is set to less than 22 symbol characters per row
-- Primary data exceeds the set width

x = Sn n = 4..22 symbol characters per row

d = 0 Normal writing direction
= 1 Printout rotated by 90°
= 2 Printout rotated by 180°
= 3 Printout rotated by 270°

w = W Counter field TEXT is incremented/decremented without a carry over, i.e. only the first unit position of the figure is increased/decreased.

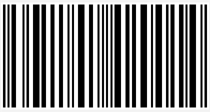
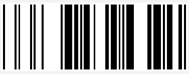
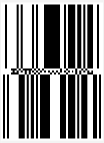
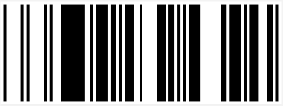
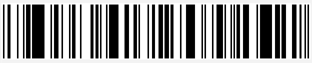
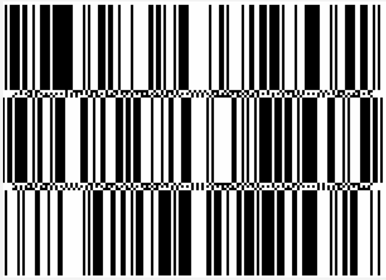
s = int Number of printer dots for the bar code module width.

v = + Incrementation – the offset is added to TEXT.
= - Decrementation – the offset is subtracted from TEXT.

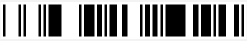
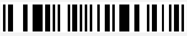
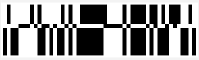
o = int Offset, which is added to (incremented) or subtracted from (decremented) the TEXT depending on the prefix.

a = int Number of labels with the same constant number (1 to 255).

TEXT Primary data and optional secondary data (separated by a „|“); alphanumerical code according to the selected bar code type, up to a maximum of 1024 characters long.

Example GS1 DataBar Testcard *)		**Symbol 1** – EAN-13 Data Output: 9501101420014 EasyPlug: #YB1/0O/16/3///9501101420014
		**Symbol 2** – GS1 DataBar Omnidirectional Data Output: 0109501101420021 EasyPlug: #RSS1/0/3///0950110142002
		**Symbol 3** – GS1 DataBar Stacked Omnidirectional Data Output: 0109501101420038 Easy-Plug: #RSS4/0/3///0950110142003
		**Symbol 4** – GS1 DataBar Omnidirectional (POS and General Distribution) Data Output: 0109501101420045 Easy-Plug: #RSS1/0/6///0950110142004
		**Symbol 5** – GS1 DataBar Expanded Data Output: 01095011014200522112345678 Easy-Plug: #RSS6/0/3///01095011014200522112345678
		**Symbol 6** – GS1 DataBar Expanded Stacked Data Output: 1095011014200693922995<GS>3202000100 17100101422123<GS>2112345678 Easy-Plug: #RSS6S6/0/3///01095011014200693922995 <FNC1>320200010017100101422123 <FNC1>2112345678#G CAUTION! No <cr><lf> in Easy-Plug sequence or Ausgabe.

A = TTX 600 --TLX 604 – Eagle **B** = TTX 300 Offline - Cobra Offline **D** = TTX 300 Online –TTX 207 - Cobra Online – Da Capo
E = TTX 450/650/674/675/950/1050 – Puma/Puma plus/Lion/Lion plus/Tiger/Tiger XXL -- S45/65/95/105 - TDI/STDI/XXTREME
G = TTX 350 – Ocelot – TTK – Texxtile **H** = 64-xx – Chess x – DPM – PEM – ALX 92x – AP 4.4 – AP 5.4 – AP 5.6 – AP 7.t

		**Symbol 7** – GS1 DataBar Truncated Data Output: 0109501101420076 Easy-Plug: #RSS2/0/3///0950110142007
		**Symbol 8** – GS1 DataBar Limited Data Output: 0109501101420083 Easy-Plug: #RSS5/0/3///0950110142008
		**Symbol 9** – GS1 DataBar Stacked Data Output: 0109501101420090 Easy-Plug: #RSS3/0/3///0950110142009

*) *The example is an instruction for printing the bar codes as they are applied in the „GS1 DataBar 2010 Test Card – V2" document (see www.gs1.org/databar_readiness/).*

#RT Read and print RFID data

Definition This command reads data of a RFID transponder and prints it on its label.

➡ Only for printers with installed and activated RFID option.
➡ Command must be placed between #ER and #Q!

Syntax #RTz/dbjk/t/ab/s/n Group H

z = int Character set number (100 to 116, refer to Info-Printout "Font Status" or refer to topic section "Internal Fonts")

d = 0 Normal write direction
= 1 Text rotated by 90 degrees
= 2 Text rotated by 180 degrees
= 3 Text rotated by 270 degrees

b = A Text printed inversely (white characters on black background):
➡ A dark background must first be applied to the print area (line or rectangle).
➡ Inverse printed offset is not possible within Groups A and B.

j = L Flush left (default) :
The write command (#T / #J) refers to the left edge of the text field. The field is built up to the right.

= M Middle:
The write command (#T / #J) refers to the centre of the text field. The field is built up on both sides of the print position.

= R Flush right:
The write command (#T / #J) refers to the right edge of the text field. The field is built up to the left.

k = Snum Fixed distance in mm between all characters in the read data string. The character S has to be succeeded without blank by the value.

t = int Data type
0 = BLOCK [1] [2]
1 = UID/TID [3]
2 = EPC
3 = KILL Password
4 = ACCESS Password
5 = MEM BANK (only read operations)

6 = AFI
7 = MEMBLOCK 1) 2)
8 = DSFID

1) „s“ required
2) „n“ required
3) “n” optional – for limiting size of data returned for read operations.

a = I ASCII data assumed on transponder (default)
Each byte on chip is printed as ASCII character on the label.

= B Binary data assumed:
Each byte chip is printed as hexadecimal number in ASCII presentation on the label.

b = L Least significant byte of block first (default) :
Bytes of each block on transponder are read beginning with the LSB first.

= M Most significant byte of block first:
Bytes of each block on transponder are read beginning with the MSB first.

s = int If Data type =
-- BLOCK: start block address
-- MEM BANK: address/name of memory bank; (default: 0)
-- MEMBLOCK: combination of memory bank identifier and address offset in the memory bank. Schema for address calculation:
s = <EPC MemoryBankID> * 1000 + <BlockNumber>
EPC MemoryBankID:
0: reserved
1: EPC
2: TID
3: USER
Default setting: 0. Other values are ignored.
-- Other data type: parameter is ignored

n = int If…
-- Data type = BLOCK, TID, MEM BANK or MEMBLOCK: number of blocks to be read (default: 0)
-- Other data type: parameter is ignored

➡ For reading a TID field (t=1), the size parameter can be used to limit the size of the returned data. Giving no size information (n=0) will return the whole content of the TID memory bank, which may contain more information than expected (depending of RFID transponder/chip). Typical values for n are:
2: Read 32Bit Vendor-ID/Chip-ID
4: Read 32Bit Vendor-ID/Chip-ID + 32Bit Unique Serial Number

Example		Data on transponder (block size 4 assumed e.g. I-Code) shown in hex notation: MSB ... LSB block address 3: 41 42 43 44 block address 4: 30 31 32 33

	#RT101////3/2	2 blocks from transponder starting at block 3 will be read and printed with font 101 in normal (from left to right) print direction. Text printed on label: DCBA3210
	#RT101///B/3/2	Now binary data Text printed on label: 443424133323130
	#RT101///M/3/2	Now Ascii data reversed Text printed on label: ABCD0123
	#RT105//2/B	EPC printed in hex notation with font 105 in normal (from left to right) print direction.

#RTC Setting the realtime clock

Definition Sets time and date of the realtime clock.

➡ Command must be placed *outside* of #ER and #Q!

Syntax	#RTC/date#G	Group H

date = dd.mm.yyyy hh:mm

(dd = day; mm = minute; yyyy = year; hh = hour; mm = minute)

Example	#RTC/01.02.2007 01:32#G

#RX Select gap sensor

Definition Switching to the required photoelectric switch by means of a command.

➡ Command must stand outside of #ER and #Q!

Syntax	#RXn	Groups E, G, H

n	=	0	Transparency
	=	1	Reflex
	=	2	Full-size

S

#S Dispensing mode/position

Definition Defines the dispensing mode and dispensing position.

This command is only for printers in the dispenser version. The single start function is not affected by it.

Syntax #Sm/p Groups D, E

m = B Batch mode:
- for materials with high separation values, i.e. not always suitable for dispensing
- higher label capacity (output). The start of the label cannot be printed over the length selected under SPOS or "p".

= E 1:1 mode: (not for Group D)
- for materials with low separation values which are suitable for dispensing
- total surface of the label is printable. The capacity is somewhat reduced because the label is pulled back to the zero line.

p = num Dispensing position (13 to 100 mm)
Distance from the back edge of the label to the zero line on the print head.

Example	#SE/14	1:1 mode and dispensing position 14 mm

#SB Bar code definition

Definition of a bar code. Printing of the bar code requires a subsequent #VW-command.

➡ Command must be placed between #ER and #Q!

Syntax #SBz/kclbmrev/h/s#G Group H [1]

z = int Bar code number (see Tab. 8).

#	Bar code	#	Bar code
0	EAN 8	14	MSI
1	EAN 13	15	EAN 128
2	UPCA	16	CODE 39 (3:1)
3	CODE 93	17	POSTCODE (guide and identity code)
4	CODE 2/5 Interleaved	18	CODE 128 (UPS)
5	CODE 2/5 Matrix	19	CODE 39 (2,5:1)
6	CODE 2/5 5-line	20	CODE 2/5 Interleaved Ratio 1:3
7	CODE 39	21	CODE 2/5 Matrix Ratio 1:2,5
8	CODABAR	22	CODE 2/5 Matrix Ratio 1:3
9	UPCE	23	CODE 39 Extended

[1] *AP 4.4: only with 16 MB RAM and firmware version 3.0 or higher.*

#	Bar code	#	Bar code
10	ADD ON 2	24	CODE 128 A
11	ADD ON 5	25	CODE 128 B
12	ITF	26	CODE 128 C
13	CODE 128	27	CODE 128 Pharmacy

Tab. 6 Bar code selection is done by inserting the appropriate number into the #SB command.

Printing a bar code list with print samples:
➔ Call up parameter PRINT INFO > Font status.

k = M Bar code with plain-copy line
= O Bar code without plain-copy line

c = C A check digit according to Module 10 is calculated and printed.
➡ Works only with 2/5 interleaved bar codes and with ITF bar code!
= N No check digit calculation

l = H Plain copy line justified
= I Plain copy line left-justified
= K Plain copy line centered
= L Plain copy line right-justified
➡ Default setting depends on the barcode type.

b = A Position change of the plain-copy line. If the plain-copy line is normally printed below the bar code, it is shifted by this option to the top of the bar code, and the other way round.

m = B EAN/UCC mode with brackets around the data designator.
The data have to be sent in brackets! The brackets appear in the plain copy line but not as bar code.

= X EAN/UCC mode without brackets around the data designator.
Data have to be sent without brackets.

r = Pnum Ratio of the bar code (2.0 to 3.0)
The letter P (proportion) must stand immediately in front of the ratio (e.g. P2.5). A ratio without the letter P is invalid.

e = V Bar code is verified.
➡ Only with connected and activated OLV.
➡ Only applicable for bar codes which are rotated by 0° or 180°.

h = int Bar code height:

Bar code height = (h + 1) * PRINT PARAMETERS > Bar code multip.

0 → 1 mm, 1 → 2 mm, 2 → 3 mm bar code height, if parameter PRINT PARAMETERS > Bar code multip. is set to 1 (default).

s = int Bar code width factor (1 to 30 Dot)

#SCF Codablock F definition

Definition of a “Codablock F” bar code. Printing of the bar code requires a subsequent #VW-command.

Command must be placed between #ER and #Q!

Syntax	#SCFs/m/c/r#G	Group H (without AP 4.4 / 8MB)

s = int Bar code width (1 to 30); Default setting: 1

m = int Height of a Codablock-row in mm (1 to 100); Default setting: 5 mm

c = int No. of columns (4 to 62); Default setting: 10

r = int No. of rows (2 to 44); Default setting: 0
0: Number of rows is calculated by codablock.
1: Improper value, interpretation causes an error message.

#SDM Data Matrix definition

Definition of a “Data Matrix” bar code. Printing of the bar code requires a subsequent #VW-command.

Command must be placed between #ER and #Q!
Not with AP 4.4 with 8 MB RAM.

Syntax	#SDMn/irc/s#G	Group H

n = int Encoding methods (details see #IDM)
0 = ASCII
1 = C40
2 = TEXT
3 = BASE256
4 = reserved
5 = AUTO (Default setting)

Encoding methods:

- ASCII
  Encoding of ASCII data, double density numeric data and symbology control characters.
- C40
  Encoding of ASCII data. Packs three alpha numerical data characters into two code words. Usage if data contains more upper case than lower case characters.
- TEXT
  Encoding of ASCII data. Packs three alpha numerical data characters into two code words. Usage if data contains more lower case than upper case characters.
- BASE256
  Encoding of any 8 bit data.

Set SYSTEM PARAMETERS > Character filter to „All characters“, if all characters from 0x00 to 0xff are supposed to be coded!

i = B EAN/UCC mode with data designator put in parantheses. Data has to be sent in parantheses, but the parantheses will not be printed coded. Same handling as EAN 128 bar code.

= X EAN/UCC mode with data designator not put in parantheses. Data has to be sent without parantheses. Same handling as EAN 128 bar code.

r = Rn n = Number of rows

c = Sn n = Number of columns

As a standard, the size (number of rows and columns) is calculated automatically in dependance of the input data (smallest possible matrix). Use the parameters r and c to set the size.

Row	Column	Row	Column
10	10	64	64
12	12	72	72
14	14	80	80
16	16	88	88
18	18	96	96
20	20	104	104
22	22	120	120
24	24	132	132
26	26	144	144
32	32	8	18
36	36	8	32
40	40	12	26
44	44	12	36
48	48	16	36
52	52	16	48

Tab. 7 Admissible combinations of rows (r) and columns (c).

s = int Number of printer dots for one Data Matrix block (1 to 60).

Example	#ER #SDM/B/14#G #T10#J5 #T10#J5 #VW/L/"(01)34012345123457(10)12345 <FNC1>(17)101231"#G #Q1/	EAN/UCC mode

#SF Fixfont definition

Defines the font for a not scalable text field (fixfont). Printing of the text field requires a subsequent #VW-command.

➡ Command must be placed between #ER and #Q!

➡ Not with AP 4.4 with 8 MB RAM.

Syntax #SFz/k/b#G Group H

z = int Font number (100 to 116)

❍ Refer to the Info-Printout PRINT INFO > Font status or refer to the user manual, topic section "Internal Fonts" for a listing of all internal fonts.

➡ In case of an invalid font number, the text is printed with font 100.

. k = Snum Fixed distance in mm between all characters in TEXT. Input in millimeters. The distance is measured from character beginning to character beginning.

➡ The character S has to be succeeded without blank by the value “num”.

b = int Micro spacing measured in printhead dots. The value is added to the standard character distance and allows fine adjustment of the text width. Value range [0…16].

#SFN Code 49 definition

Definition of a “Code 49” bar code. Printing of the bar code requires a subsequent #VW-command.

➡ Command must be placed between #ER and #Q!

Syntax #SFNm/kx/s/h#G Group H (without AP 4.4 / 8MB)

m = 0 Alphanumeric Mode
= 1 Append Mode
= 2 Numeric Mode
= 3 Group Alphanumeric Mode
= 4 Alphanumeric Mode, Shift 1
= 5 Alphanumeric Mode, Shift 2
= 6 Reserved
= 7 Automatic Mode (default setting). The printer determines starting mode and encodation method by analyzing TEXT. This is the recommend mode.

k = M Bar code with plain copy line
➡ The plain copy line can extendet beyond the right edge of the code!
= O Bar code without plain copy line (default setting)

x = J Plain copy line below the bar code
= A Plain copy line above the bar code

h = int Row height

Row height = (h + 1) * PRINT PARAMETERS > Bar code Multip.

#SG Graphics definition

Definition of a graphic. Printing of the bar code requires a subsequent #VW-command.

➠ Command must be placed between #ER and #Q!

Syntax	#SG#G	Group H (without AP 4.4 / 8MB)

Example	#SG#G #T82#J75 #VW/L/"Graphic01.bmp" #G	Graphics definition Define print position Print graphics

#SI Write data to interface

Trigger point definition for context interface (Easy-Plug interface).

This command is useful for many applications, for example logging of printed labels or soft “realtime” controls over the interface.

Writing data to the current Easy-Plug interface is done by the #VW/I/expression command. The point of time for interface write (trigger point, see Fig. 6 on next page) can be defined by the #SI command.

Syntax	#SIr/n#G	Group H (without AP 4.4 / 8MB)

r = int Logical point in time concerning print control that defines when to send data (trigger point, see Fig. 6 on next page).

1 Start of a new Easy-Plug format
2 Restart of a previous Easy-Plug format (#Q without #ER)
3 Start of a label execution
4 Start of RFID operation
5 End of successful RFID operation(s); if read/write operations fail, the trigger doesn’t fire
6 Start of the real print process
7 End of the real print process (executed only, if label is printed without error)
8 End of a label execution (executed only, if label is executed without error)
9 End of an Easy-Plug format (#Qn label printed)
10 Any motion control error
11 Any RFID error (tag read or tag write error); triggered together with the bad tag signal

➠ If a motion control error happens during label execution, possible later trigger operations are not executing.

➠ RFID: To transmit data, which were read from or written to the transponder, it is recommended to use trigger point r = 8. This makes sure that RFID operations and printing have been successfully completed.

➠

n = int	Number of bytes to respond
	n <> 0: The return string (defined with #VW) is truncated to byte count n. If the respond string counts less than n bytes, it is filled with blanks (hex code 0x20).
	n = 0: The size of the respond string is whatever is necessary to send the requested data (default).

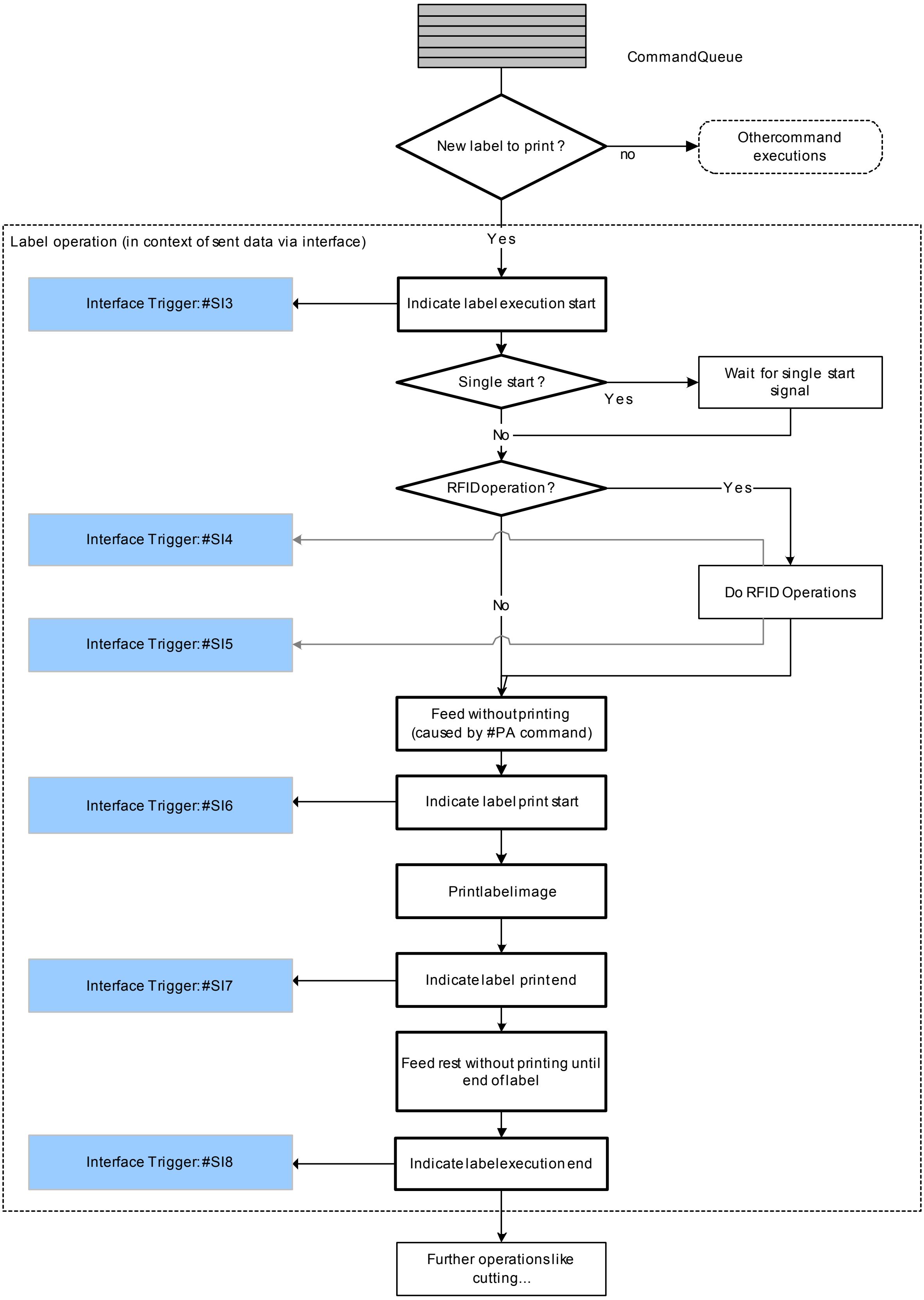


Fig. 6 Flowchart showing label operation trigger points #3 - #8.

Example 1
Simple example

```
#G -----------------------------------------------------------------------------------------
#G Returns the printer counter value, if the label is printed without error
#G -----------------------------------------------------------------------------------------
#!A1
#IMN100/100
#ER
#G --------------------------------------------------
#G Define and print counter as text
#G --------------------------------------------------
#VDT/Counter/1/1/0000#G
#SF109/
#FD0
#T10#J20
#VW/L/Counter#G
#G --------------------------------------------------
#G Return counter value to interface, as soon as a label is printed
#G without error.
#G --------------------------------------------------
#SI8/#G
#VW/I/chr(10) + chr(13) + "CNT: " + Counter#G
#Q3/
#G -----------------------------------------------------------------------------------------
#G Response at active interface:
#G CNT: 0000
#G CNT: 0001
#G CNT: 0002
#G -----------------------------------------------------------------------------------------
```

Example 2
More complex example

```
#G -----------------------------------------------------------------------------------------
#G Shows the usage of more different trigger events
#G -----------------------------------------------------------------------------------------
#!A1
#IMN100/100
#ER
#G --------------------------------------------------
#G Error code return on interface in case of any print related error
#G --------------------------------------------------
#VDS/ErrorCode//M1014#G
#SI10/#G
#VW/I/chr(10) + chr(13) + "  Error code: " + ErrorCode #G
#G --------------------------------------------------
#G Easy-PLug format related messages
#G --------------------------------------------------
#SI1/#G
#VW/I/chr(10)+chr(13)+chr(10)+chr(13) +
"*** New Easy-Plug format start *** "#G
#SI2/#G
#VW/I/chr(10)+chr(13)+chr(10)+chr(13) +
"*** Easy-Plug format restart *** "#G
```

```
#SI9/#G
#VW/I/chr(10) + chr(13) + "*** Easy-Plug format finished *** "#G
#G -------------------------------------------------
#G Label related messages
#G -------------------------------------------------
#SI3/#G
#VW/I/chr(10)+chr(13) + "  * Label operation start"#G
#G -------------------------------------------------
#G Define and print counter as text
#G -------------------------------------------------
#VDT/Counter/1/1/0000#G
#SF109/
#FD0
#T10#J20
#VW/L/Counter#G
#G -------------------------------------------------
#G Return conuter value to interface, as soon a label is printed
#G without error.
#G -------------------------------------------------
#SI8/#G
#VW/I/chr(10) + chr(13) + "    Printed Counter: " + Counter#G

#VDT/TextVar///Text1#G
#SF109/
#FD0
#T10#J30
#VW/L/TextVar#G
#SI8/#G
#VW/I/chr(10) + chr(13) + "    Printed TextVar: " + TextVar#G
#G -------------------------------------------------
#G Label operation finished message
#G -------------------------------------------------
#VDD/Time/AU//^h:^m.^s#G
#SI8/#G
#VW/I/chr(10)+chr(13) + "  * Label operation finished at " +Time#G
#Q2/

#SV/TextVar/New text#G
#Q1/

#G ---------------------------------------------------------------------------------------------
#G Response at active interface:
#G *** New EasyPlug format start ***
#G  * Label operation start
#G    Printed Counter: 0000
#G    Printed TextVar: Text1
#G  * Label operation finished at 13:07.07
#G  * Label operation start
#G    Printed Counter: 0001
#G    Printed TextVar: Text1
#G  * Label operation finished at 13:07.08
```

```
#G *** Easyplug format finished ***
#G
#G *** Easyplug format restart ***
#G  * Label operation start
#G    Printed Counter: 0002
#G    Printed TextVar: New text
#G  * Label operation finished at 13:07.09
#G *** Easyplug format finished ***
#G ------------------------------------------------------------------------------------------
```

#SMX Maxicode definition

Definition of a “Maxicode” bar code. Printing of the bar code requires a subsequent #VW-command.

Command must be placed between #ER and #Q!

Syntax #SMXz/x/y#G Group H (without AP 4.4 / 8MB)

z	=	int	MaxiCode modes 2, 3, 4 and 6
x	=	int	Symbol number for code-splitting (1 to 8)
y	=	int	Number of symbols for code-splitting (1 to 8)

#SPF PDF 417 definition

Definition of a “PDF 417” bar code. Printing of the bar code requires a subsequent #VW-command.

Command must be placed between #ER and #Q!

Syntax #SPFnt/s/l/z/w/h#G Group H (without AP 4.4 / 8MB)

n	=		Compression type
	=	0	EXC mode (Extended alphanumeric Compaction Mode)
	=	1	Binary ASCII Plus Mode
t	=	T	Unidirectional reading
	=	(blank)	Bi-directional reading
s	=	int	Security Level (0 bis 8)
l	=	0	Bar code width is set automatically
	=	int	Bar code width in code words (number of columns) (1 to 30): The width should be so selected that a height of 90 lines is not exceeded with regards to the area of the bar code defined in the user data. A maximum of 928 code words are permitted for one bar code. Of these the following are preassigned: – 1 word for the length input of the code – x words for the check total (Security Level) = 2 (1 + Security Level)
z	=	0	Number of lines is set automatically
	=	int	Number of lines (3 to 90)
w	=	int	Bar code width (1 to 16)
h	=	num	Height of a PDF bar code row in millimetres (0.25 to 100.0)

#SQR QR Matrix Code Definition

Definition of a “QR matrix code” bar code. Printing of the bar code requires a subsequent #VW-command.

➡ Command must be placed between #ER and #Q!

Usage pattern:

1. Define data in variable:	#VDT/QR_Data////qr code#G
2. Set context parameters:	#SQR2/LU/6///#G #FD/0/L#G#T5.0#J10.0
3. Write to media context “label”:	#VW/L/QR_Data#G

Syntax #SQRm/ei/s/an/d/p#G Group H (without AP 4.4 / 8MB)

m =	int	1: QR matrix code model 1 2: QR matrix code model 2 (default)
		Error correction levels:
e =	L	7% High density
=	M	15% Standard (default)
=	Q	25% High reliability
=	H	30% Ultra high reliability
		Character sets:
i =	A	Automatic (input data type is selected automatically; default)
=	U	User defined (input data type is defined by the user)

Valid input data types are:

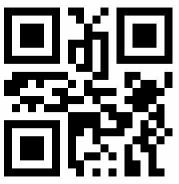
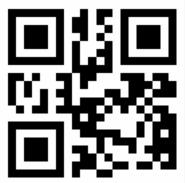
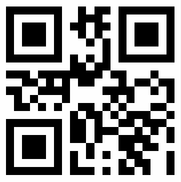
N for numeric data (digits 0 - 9)
A Alphanumeric data (digits 0 - 9; upper case letters A -Z; nine other characters)
B Binary data. The number of bytes must be specified after the ‘B’ by a 4 digit number indecimal (ex. B0003123#G)
K Kanji characters. Kanji characters in QR Code can be compacted into 13 bits

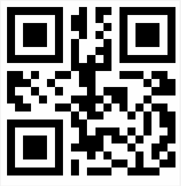
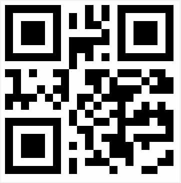
➡ The character indicating the input data type must be put right in front of the input data.

s =	int	Module size in pixel (at least 4; 4 is default)
a =	S	Single symbol (default)
=	A	Enables „Structured Append“
		Symbol Sequence number:
n =	int	1 up to number of divisions (valid if Structured Append is enabled)
		Total number of symbols:
d =	int	2 up to 16 (valid if Structured Append is enabled)

Parity:

p =	int	0 up to 255 (valid if Structured Append is enabled)

Example	#!A1 #IMN100.0/100.0// #N10 #ERN// #VDT/QR_Data_StructuredAppend_Part1////AAAAAAAA#G #VDT/QR_Data_StructuredAppend_Part2////BBBBBBBB#G #VDT/QR_Data_StructuredAppend_Part3////CCCCCCCC#G #VDT/QR_Data_StructuredAppend_Part4////DDDDDDD#G #VDT/QR_Data_Manual////N0123456789,A---123-ABC %$+-/:,B0004ÿÿÿ#G #VDE/QR_Data_4PartsInOne/QR_Data_StructuredAppend_Part1+ QR_Data_StructuredAppend_Part2+QR_Data_StructuredAppend_Part3+ QR_Data_StructuredAppend_Part4#G #FD/0/L#G	
	#SQR2/QU/4///#G #T40.0#J80.0 #VW/L/QR_Data_Manual#G	
	#G ------------------------------------- #G Binary data "Test<cr><lf>" #G ------------------------------------- #SQR2/QU/4///#G #T20.0#J40.0 #VW/L/"B0006Test"+CHR(10)+CHR(13)#G	
	#G ------------------------------------- #G All in one #G ------------------------------------- #SQR2/LA/4///#G #T50.0#J25.0 #VW/L/QR_Data_4PartsInOne#G	
	#G ------------------------------------- #G Structured Append 1..4 #G ------------------------------------- #SQR2/LA/4/A1/4/255#G #T20.0#J10.0 #VW/L/QR_Data_StructuredAppend_Part1#G	
	#SQR2/LA/4/A2/4/255#G #T40.0#J10.0 #VW/L/QR_Data_StructuredAppend_Part2#G	

	#SQR2/LA/4/A3/4/255#G #T60.0#J10.0 #VW/L/QR_Data_StructuredAppend_Part3#G	
	#SQR2/LA/4/A4/4/255#G #T80.0#J10.0 #VW/L/QR_Data_StructuredAppend_Part4#G #Q1#G	

#SRF RFID Read/Write Definition

Defines the target (write) or source field and address (read) of any following #VW/T or #VR/T command.

Command must be placed between #ER and #Q!

Syntax #SRFt/bfs/n/r#G Group H (without AP 4.4 / 8MB)

t = int

Data type
0 = BLOCK [1) 2)]
1 = UID/TID [3)]
2 = EPC
3 = KILL Password
4 = ACCESS Password
5 = MEM BANK (only read operations)
6 = AFI
7 = MEMBLOCK [1) 2)]
8 = DSFID

1) „s“ required
2) „n“ required
3) “n” optional – for limiting size of data returned for read operations.

b = L

(Default) The bytes of the expression value used in the #VW/T resp. #VR/T command are written to the transponder with the least significant (LSB) byte first.

= M

Same as above, with the most significant byte (MSB) first.

f = P

Write protection

s = int

If Data type =
-- BLOCK: start block address
-- MEM BANK: address/name of memory bank; (default: 0)
-- MEMBLOCK: combination of memory bank identifier and address offset in the memory bank. Schema for address calculation:
s = <EPC MemoryBankID> * 1000 + <BlockNumber>

EPC MemoryBankID:
0: reserved
1: EPC
2: TID
3: USER
Default setting: 0. Other values are ignored.

-- Other data type: parameter is ignored

n = int If…

-- Data type = BLOCK, TID, MEM BANK or MEMBLOCK: number of blocks to be read (default: 0)

-- Other data type: parameter is ignored

➡ For reading a TID field (t=1), the size parameter can be used to limit the size of the returned data. Giving no size information (n=0) will return the whole content of the TID memory bank, which may contain more information than expected (depending of RFID transponder/chip). Typical values for n are:
2: Read 32Bit Vendor-ID/Chip-ID
4: Read 32Bit Vendor-ID/Chip-ID + 32Bit Unique Serial Number

r = int Number of read/write retries (default setting = setting in RFID PARAMETER > Nr CMD retries).

INFO: Reading from / writing to a RFID tag

Reading data from and/or writing data to a RFID tag always requires a variable and two explicit commands:

1. Define a variable: #VDT
2. Define target field or data type: #SRF
3. Read or write data: #VR or #VW

#SRS GS1 DataBar definition

Definition of a “GS1 DataBar” (formerly “RSS”) bar code. Printing of the bar code requires a subsequent #VW-command.

➨ Command must be placed between #ER and #Q!

Syntax	#SRS/zt/s#G	Group H (without AP 4.4 / 8MB)

z = int Bar code number

❍ See topic section „Info-printouts & Parameters“, paragraph „Font status“, Bar code library“.

➨ *GS1 DataBar Expanded Stacked* is applied automatically, if the following conditions are met:
-- Selected barcode type: GS1 DataBar Expanded (z = 6)
-- Width (t) is set to less than 22 symbol characters per row
-- Primary data exceeds the set width

t = Sn Width in segments; value range: n = 4…22

➨ If primary data exceeds the width in segments, the code is stacked automatically.

s = int Number of printer dots for the bar code module width.

#SS Speedo font definition

Defines the font for a scalable text field (speedo font). Printing of the bar code requires a subsequent #VW-command.

➨ Command must be placed between #ER and #Q!

Syntax	#SSz/ghrkp/ex/s #G	Group H (without AP 4.4 / 8MB)

z = int Font number

➨ The fonts 100 and 101 are internal scalable fonts and are therefore always available.

❍ See printer manual, topic section "Internal Fonts".

g = O Default type
= B Bold

h = V Default type
= I Italics

r = K Application of scalable fonts without Unicode (default setting)
= U Application of Unicode fonts
= G Application of arabic Unicode fonts (reading direction right to left)

k = Snum Fixed distance in mm between all characters in TEXT.

➨ The character S has to be succeeded without blank by the value “num”.

p = N (Default) Standard character-algorithm is applied.
Q 2 D character algorithm is used. May improve the exactness of small characters. (e.g. printing of small chinese characters). This algorithm is about 30-40% slower than the standard algorithm.

e = int Font size in dot in y-direction, depending on the font and its installed size, respectively (see example below). Value range: [5…6000] Dot.

x = Xint Font size in dot in x-direction, depending on the font and its installed size, respectively (see example below). Value range: [5…6000] Dot.

➡ The character X has to be succeeded without blank by the value “int”.

s = int Micro spacing measured in printhead dots. The value is added to the standard character distance and allows fine adjustment of the text width. Value range [0…16].

#SV Changing the content of a text variable

➡ Command is allowed between or outside #ER and #Q!

Syntax	#SV/name/TEXT#G	Group H (without AP 4.4 / 8MB)

name = Variable name (must be defined with #VDT before!)

TEXT = New value of the variable (max. length: 4096 characters)

T

#T Horizontal print position

Definition Determines the horizontal print position with absolute value in mm.

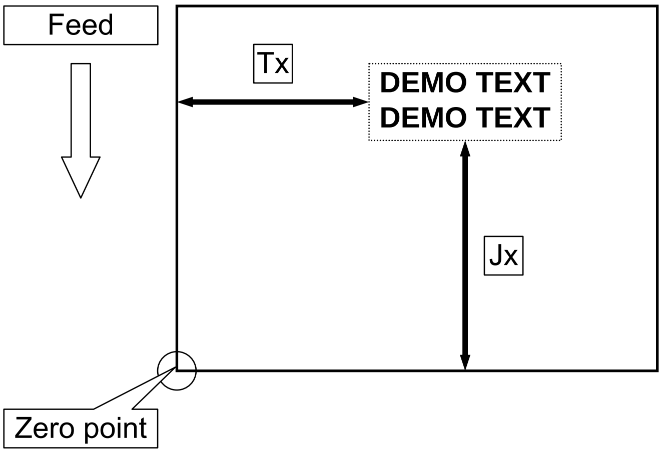
➠ The horizontal zero position is always the position of the first dot of the printhead. Looked at in feed direction, this dot is placed at the right end of the printhead. Exception: The zero position was shifted with the #R command.

If the printhead is positioned at the inner limit (right end if viewed in feed direction), the first dot has a distance of 1 mm to the material zero line. In other words: Looked at in feed direction there is a 1 mm wide stripe at the right end of the label, which stays unprinted.

➠ Command must stand between #ER and #Q!

Syntax	#Tx	All Groups

x = num Horizontal print position in mm in relation to the zero position (see above)



➠ Text outside of the defined label area is not printed.

Example	#T5	Text begins 5 mm from the left
	#T20	Text begins 20 mm from the left

V

#VDT Define text variable

Definition Definition of a text variable with counter.

➡ The command must be placed between #ER and #Q!

Syntax #VDT/name/wz/vo/a/TEXT#G Group H (without AP 4.4 / 8MB)

name = Variable name

➡ Allowed are alphanumerical characters without the following:

Char.			!	"	#	,	.	+	-	*	/
Name		blank	exclam	quotedbl	numbersign	comma	period	plus	hyphen	asterisk	slash
ANSI	<32	32	33	34	35	44	46	43	45	42	47
Hex	<20	20	21	22	23	2c	2e	2b	2d	2a	2f

w = W Counter field TEXT is incremented/decremented *without* a carry over, i.e. only the first unit position of the figure is increased/decreased.

= C Counter field TEXT is incremented/decremented *with* a carry over

z = Z Leading zeros in counter fields are printed.

= S Suppresses leading zeros in counter fields. The remaining digits are printed at the same position they were printed at *with* leading zeros.

v = + Incrementation – offset is added to the text field.
= - Decrementation – offset is subtracted from the text field.

o = int Offset, which is added to (incremented) or subtracted from (decremented) the TEXT depending on the prefix.

a = int Number of labels with the same constant number (1 to 255).

TEXT = Any alphanumerical text, whereby only numerical values are taken into account during offset. All numerical values in the text string are taken into account for the offset (e.g. 10A3B56 = 10356). Maximum number of characters: 255.

#VDD Define date and time variable

Definition Definition of a variable which can contain date and/or time.

➡ The device must be equipped with a realtime-clock.
➡ The command must be placed between #ER and #Q!

Syntax #VDD/name/uv/o/TIMETEXT#G Group H (without AP 4.4 / 8MB)

name = Name of the variable (admissible characters see page 3).

u = (void) as u = W.

= W (Default setting) No update. The time is read into the variable *once*. This is done during the compilation of the layout.

= U The time is read repeatedly during the running print job. This happens immediately after printing the relevant output field (#VW…), but counts only for the same field on the *next* label.

➡ The setting is not suitable for dispensers.
Reason: If the next label is printed considerably later (e.g. in single-start mode with footswitch), the following happens: The output field on label 2 contains the time, at which the same field on label 1 was printed.

Example Single-start mode	Label 1 is printed at 10.50 h, label 2 at 11.03 h. In this case, the imprint of label 2 shows the time 10.50!

= Y (Recommended for dispensers) The time is read repeatedly during the printing of a label (approx. all 5 seconds).

v = A Different times may be printed on a label (if option U or Y is applied).

o = (blank) The current date is printed.
= 0 The current date is printed.
= int Days offset: The current date is calculated forward by the entered number of days (1 to 65000).
= Mint Months offset: The current date is calculated forward by the entered number of months (1 to 65000).
(Adding the letter M to the integer value changes the offset from day to month!)

TIMETEXT = String containing Time and Text (ASCII-characters) in any order. Times are marked with the control character (^). Insert required Text as ASCII characters (without control characters) in the TIMETEXT-string.

Time formats:

= ^ z 1/100 seconds (not with group H: "^ z" always leads to "00" printout)
= ^ s seconds
= ^ m minutes
= ^ h hours
= ^ D day of the month
= ^ d (Only for group H) day of the year
= ^ W 3-figure day (e.g. 001 for the 1st day of the year)
= ^ M month
= ^ Y year, two digit

=	^ R	year, four digit
=	^ G	3-figure month, German
=	^ E	3-figure month, English
=	^ S	Name of a month from command #DM
=	^ C	(Only group H) Week of the year always with two digits
=	^ c	(Only group H) Week of the year without leading 0
=	^ K	(Only group H) Corresponding year to the week (of the year) with 4 digits
=	^ k	(Only group H) Corresponding year to the week (of the year) with 2 digits

#VDE Define expression variable

Definition Definition of a variable, to which an expression can be assigned.

➠ The command must be placed between #ER and #Q!

Syntax	#VDE/name/o/expression #G	Group H (without AP 4.4 / 8MB)

name =	Name of the variable (admissible characters see page 3).
o =	Not applied option for future firmware versions.
expression =	An expression can contain constant values, variables and functions

❍ See topic section “Special Applications”, chapter Expressions 🗋.

Example

```
#G -------------------------------------------------------------
#G “(00)” = Text
#G „+“ = Operator
#G „NVE“ = Variable
#G MOD10(NVE) = Function MOD10 with variable NVE as argument
#G -------------------------------------------------------------
#VDE/BarcodeData//“(00)“+NVE+MOD10(NVE)#G
```

#VDS Define system variable

Definition Definition of a system variable, to which can be assigned name and value of each parameter in the printer menu.

➠ The command must be placed between #ER and #Q!

Syntax	#VDS/name/format/RemoteID#G	Group H (without AP 4.4 / 8MB)

name	=		Name of the variable (admissible characters see page 3).
format			Format of the system variable. The format can consist of the following placeholders:
	=	%N	Parameter name
	=	%S	Parameter value as string (equals the string on the printer display)
	=	%i	Parameter value as integer value (precondition for „%i“: the parameter must be of the integer type)
RemoteID	=	int	ID-no. of the parameter

❍ A listing of all available ID-numbers can be generated by calling SPECIALFUNCTION > Parameter to CF).

❍ Example printout of a parameter listing: user manual, topic section „Advanced Applications“, “Appendix”.

❍ See also command #!PG.

Example	Parameter:	SYSTEM PARAMETERS > Temp. reduction = „20%“
	RemoteID:	2026
	%N	Temp. reduction
	%S	Temp. reduction 20%
	%i	20

#VR Read RFID Data

➠ The command must be placed between #ER and #Q!

Syntax	#VR/context/expression#G	Group H (without AP 4.4 / 8MB)

context = T Data are read from the RFID-transponder.

expression = An expression can contain constant values, variables and functions

❍ See topic section “Special Applications”, chapter Expressions 🗋.

#VTS Define standalone variable

Definition Defines a variable, which gets a value in standalone mode via the printer operation panel or an external keyboard.

➠ The command must be placed between #ER and #Q!

Syntax	#VTS/name/wzy/l/vo/a/TEXT#G	Group H (without AP 4.4 / 8MB)

name =		Name of the variable (admissible characters see page 3).
w =	W	Counter field TEXT is incremented/decremented *without* a carry over, i.e. only the first unit position of the figure is increased/decreased.
=	C	Counter field TEXT is incremented/decremented *with* a carry over
z =	Z	Leading zeros in counter fields are printed.
=	S	Suppresses leading zeros in counter fields. The remaining digits are printed at the same position they were printed at *with* leading zeros.
y =	U	Update of the standalone input field. After each run of the printjob, the latest counter value appears as default on the printer display.
=	N	*No update* of the standalone input field. After each run of the printjob, the default counter value appears as default on the printer display.
l =	int	Maximum length of the input field on the operation panel in standalone mode.
v =	+	Incrementation – offset is added to the text field.
=	-	Decrementation – offset is subtracted from the text field.
o =	int	Offset, which is added to (incremented) or subtracted from (decremented) the TEXT depending on the prefix.
a =	int	Number of labels with the same constant number (1 to 255).

TEXT = Any alphanumerical text, whereby only numerical values are taken into account during offset. All numerical values in the text string are taken into account for the offset (e.g. 10A3B56 = 10356). Maximum number of characters: 255.

#VW Drawing/writing to target

Definition Sends the content of the expression to a target.

➠ The command must be placed between #ER and #Q!

Syntax #VW/context/expression#G Group H (without AP 4.4 / 8MB)

context = T Target = RFID tag, what means that the value of the expression is sent to a RFID transponder.

= I Target = Easy-Plug interface, what means that the expression value is sent to the interface, which is defined in INTERF.PARAM. > EASYPLUGINTERPR. > Interface.

= L Target = Label, what means that the expression value is printed on a label.

expression = An expression can contain constant values, variables and functions

❍ See topic section “Special Applications”, chapter Expressions 🗋.

Y

#Y./.F./. Data exchange

Function

With the switch <F>, user data outside of the format #ER to #Q/ can be sent and assigned to the preceding format.

It is not necessary to transfer the entire format if the user data is altered. Only the position, font and altered user data is sent. Duplicate data is not sent, which considerably reduces transmission times.

Definition

The switch <F> can be used in conjunction with the commands #YT, #YB, #YC and #YS. Switch <F> can be set anywhere between the first two slashes of the preceding #Y command.

#Y commands marked with <F> are deleted by a following #Y command outside of the format.

The required position (#J, #T), font number, display options and user data must be defined with the following #Y command. The quantity must also be entered.

➠ The command #Y should not be used for printers belonging to the E and G groups. Use the Variable Data Field command (#YV) instead.

Syntax	#Y./.F./.	Groups D, E, G

Example	#!A1 #IMSB100/80 #ERN// #J5#T10#YT102//// Article number	Data about a label on which the "article number" also remains constant as fixed text for subsequent print jobs. The number designation itself changes after each print job.
	#J5#T30#YT102/F///284-BC #Q10/#G	Field which is deleted by a following #Y command outside of the format.
	#J5#T30#YT102/F///367-AC #Q20/#G	The number is changed, all other user data and information about the label remain.
	#J5#T30#YT102/F///189-ZY #Q3/#G	Number is changed as before.

#YB Bar code definition

Definition

This command is used for (optional) parameter settings and entering text.

The decimal position of the check digit must be taken into consideration with bar codes with a consecutive number (parameter "o").

➠ Command must stand between #ER and #Q!

Syntax

#YBz/dkc/h/s/vo/a/TEXT#G Groups A, B
#YBz/dkcjfg/h/s/vo/a/TEXT#G Groups E, G, TTX 300/Cobra Onl.
#YBz/dkcf/h/s/vo/a/TEXT#G TTX 207/Da Capo
#YBz/dkbcejgrwlm/h/s/vo/a/TEXT#G Group H

z = int Bar code number (see Tab. 8).

#	Bar code	#	Bar code
0	EAN 8	14	MSI
1	EAN 13	15	EAN 128
2	UPCA	16	CODE 39 (3:1)
3	CODE 93	17	POSTCODE (guide and identity code)
4	CODE 2/5 Interleaved	18	CODE 128 (UPS)
5	CODE 2/5 Matrix	19	CODE 39 (2,5:1)
6	CODE 2/5 5-line	20	CODE 2/5 Interleaved Ratio 1:3
7	CODE 39	21	CODE 2/5 Matrix Ratio 1:2,5
8	CODABAR	22	CODE 2/5 Matrix Ratio 1:3
9	UPCE	23	CODE 39 Extended
10	ADD ON 2	24	CODE 128 A
11	ADD ON 5	25	CODE 128 B
12	ITF	26	CODE 128 C
13	CODE 128	27	CODE 128 Pharmacy

Tab. 8 Bar code selection is done by inserting the appropriate number into the #YB command.

❍ Info-printout of a bar code list with print samples: See topic section „Info-Printouts and Parameters“, parameter INFO > STA2.

→ How to print a bar code list with print samples: Call up parameter PRINT INFO > Font status (group H), or INFO > STA2 (groups E, G), respectively.

d = 0 Bar code in normal write direction
= 1 Bar code rotated by 90 degrees
= 2 Bar code rotated by 180 degrees
= 3 Bar code rotated by 270 degrees

k = M Bar code with plain-copy line
= O Bar code without plain-copy line

b = A Position change of the plain-copy line. If the plain-copy line is normally printed below the bar code, it is shifted by this option to the top of the bar code, and the other way round.

c = C A check digit is calculated and printed.

➡ Works only with bar codes with *optional* check digit.

e = V Bar code is verified.

➡ Only with connected and activated OLV.

➡ Only applicable for bar codes which are rotated by 0° or 180°.

Example Verifying bar code	#!A1 #IMS100/100 #ER #T5#J5 #YT109/0///OLV Test #OLVD/ /70///	
	#T5#J50 #YB0/0V/4/4///7777777	Bar code is verified (Decodability >= 70)
	#T5#J30#YB0/0/8/8///1234567 #OLVD/ /60//30/	Bar code is not verified
	#T5#J70 #YB0/0V/8/8///8888888	Bar code is verified (decodability >= 70 and defects <= 30)
	#Q1/	

j = Z Centered:
The field is built up on the left and right in the centre of the print position.

= R Flush right:
The field is built up flush right to the print position. The field is built up to the left.

f = F Field designation for changing data
(see description of changing data)

g = D Text field consists of one variable data field

r = Pnum Ratio of the bar code (2.0 to 3.0)
The letter P (proportion) must stand immediately in front of the ratio (e.g. P2.5). A ratio without the letter P is invalid.

➡ Observe the sequence: The ratio must be entered after the parameter d (write direction)!

w = W Counter field TEXT is incremented/decremented without a carry over, i.e. only the first unit position of the figure is increased/decreased.

l = H Plain copy line justified (default setting)
= I Plain copy line left-justified
= K Plain copy line centered
= L Plain copy line right-justified

m = B EAN/UCC mode with brackets around the data designator.
The data have to be sent in brackets! The brackets appear in the plain copy line but not as bar code.

= X EAN/UCC mode without brackets around the data designator.
Data have to be sent without brackets.

h = int Bar code height:

- Group H:
  a) PRINT PARAMETERS > Bar code multip. = 1 (default setting):

  Bar code height = h

  0 → 1 mm, 1 → 1 mm, 2 → 2 mm, 3 → 3 mm bar code height

  a) PRINT PARAMETERS > Bar code multip. > 1:

  Bar code height = (h + 1) * PRINT PARAMETERS > Bar code multip.

- Groups E and G:

  Bar code height = (h + 1) * PRINT PARAMETERS > Bar code multip.

  0 → 1 mm, 1 → 2 mm, 2 → 3 mm bar code height, if parameter PRINT PARAMETERS > Bar code multip. is set to 1 (default).

- TTX 207/300/480, Mustang 900/1000, Cobra and S 20/30:

  Bar code height = h + 1

  The h-value equals the bar code height (in milimeters):
  0 → 1 mm, 1 → 2 mm, 2 → 3 mm etc.

- 0 = 5 mm, 1 = 10 mm, 2 = 15 mm,.... from 10 = actual height in mm (e.g. 35 = 35 mm high). This height plus 1 is multiplied by the height zoom factor in parameter PRTP > BCHI set at the printer.

- All other printer groups:

  0 → 5 mm, 1 → 10 mm bar code height; if h >= 10, the h-value equals the bar code height in milimeters (e.g. 35 → 35 mm bar code heigth); Precondition: parameter PRTP > BCHI is set to 1 (default).

  Bar code height = (h + 1) * 5 * PRTP > BCHI (if h < 10)
  Bar code height = (h + 1) * PRTP > BCHI (if h >= 10)

s = int Bar code width factor in Dot:

Groups A, B, D, F	1 to 8
Groups E, G	1 to 16
Group H	1 to 30

v = + Incrementation – offset is added to the text field.
= - Decrementation – offset is subtracted from the text field.

o = int Offset, which is added to (incremented) or subtracted from (decremented) the TEXT depending on the prefix.

a = int Number of labels with the same constant number (1 to 255).

TEXT = Any alphanumerical text. The stipulations for the respective bar code must be taken into consideration. Max. number of characters: 255.

➡ The text field may alse be a *variable data field*. Precondition: the D-flag must be set.
❍ Syntax description: See chapter Variable Data Fields 🗋 on page 3.

➡ (Only group H) The text field may contain an *input field*.
❍ Syntax description: See chapter Input Fields 🗋 on page 4.

#G = The command #YB must be closed with #G.

Examples

```
#G ------------------------------------------------------------------------------------------
#G UPCA with plain-copy line, normal print direction, 3 mm high (only
#G valid with a group H printer), width 2, without offset.
#G ------------------------------------------------------------------------------------------
#YB2/0M/2/2///1234567890123#G

#G ------------------------------------------------------------------------------------------
#G EAN 8 and ADD on 5 are positioned in rows one after the other.
#G ------------------------------------------------------------------------------------------
#YB0/0/////12345678#YB11/0/////54321#G

#G ------------------------------------------------------------------------------------------
#G Bar code number 1 with variable data field, numbers 01 and 12,
#G alphanumerical characters.
#G ------------------------------------------------------------------------------------------
#YB1/0D/10/3///$01,12#G

#G ------------------------------------------------------------------------------------------
#G EAN/UCC mode with data designator in brackets (here: 10).
#G ------------------------------------------------------------------------------------------
#YB15/0B/2/3///(10)Charge1#G

#G ------------------------------------------------------------------------------------------
#G EAN/UCC mode without brackets around the date designator
#G (here: 10).
#G ------------------------------------------------------------------------------------------
#YB15/0X/2/3///10Charge1#G

#G ------------------------------------------------------------------------------------------
#G Code 128 Pharmacy
#G xxxxxxx = 7-digit CNK number (Pharmacy Product Ident Number)
#G yyyyyyyy = 8-digit sequential number for the first code.
#G ------------------------------------------------------------------------------------------
#YB27/0M/10/3/o/a/xxxxxxxyyyyyyyy#G
```

❍ For additional information about using the #YB command, refer to topic section "Bar code information".

#YC Real time as text

Definition

A time can be defined in text format using this command. The printer must be equipped witha real-time clock. Normal text can also be entered in conjunction with the time.

Any number of #YC commands are possible in the format.

➠ However, per print job only up to 3 commands (#YC or #YS) can be assigned the option U (update during the print job).

➠ The command #YC can only be used with internal *fixfonts*.

The TIME+ text string is printed with up to 90 characters.

(^^), (##)in the string: control characters (^) and hashes (#) are printed as ASCII characters.

➠ Command must stand between #ER and #Q!

Syntax		
	#YCz/dbju/o/TIMETEXT	Groups A, B
	#YCz/dbjfu/o/TIMETEXT	Groups E, G, TTX 207/Da Capo
	#YCz/dbqjuk/o/TIMETEXT	Group H without AP 4.4

z = int Character set number (see Font Library)

d = 0 Text in normal write direction
= 1 Text rotated by 90 degrees
= 2 Text rotated by 180 degrees
= 3 Text rotated by 270 degrees

b = A Text printed inversely:
A dark background must first be applied to the print area (line or rectangle). It is not possible to print a time within Group A and B which is being continuously updated.

q = E Existing Bitimage is inverted.

❍ See example 1.

j = M Middle:
The write command (#T/#J) refers to the centre of the text field. The field is built up on both sides of the print position.
= R Flush right:
The write command (#T/#J) refers to the right edge of the text field. The field is built up to the left.

u = (void) as u = W.

= W (Default setting) No update. The time is read into the variable *once*. This is done during the compilation of the layout.
= U The time is read repeatedly during the running print job. This happens immediately after printing the relevant output field (#VW…), but counts only for the same field on the *next* label.

➠ The setting is not suitable for dispensers.
Reason: If the next label is printed considerably later (e.g. in single-start mode with footswitch), the following happens: The output field on label 2 contains the time, at which the same field on label 1 was printed.

Example: Label 1 is printed at 10.50 h, label 2 at 11.03 h. In this case, the imprint of label 2 shows the time 10.50!

= Y (Only group H - recommended for dispensers) The time is read repeatedly during the printing of a label (approx. all 5 seconds).

k = Snum Fixed distance in mm between all characters in TEXT. The character S has to be succeeded without blank by the value.

f = F (Only group E) Field designation for exchanging data
(see description of exchanging data)

o = (blank) The current date is printed.
= 0 The current date is printed.
= int Days offset: The current date is calculated forward by the number of days entered (1 to 65000).
= Mint (Only group H) Months offset: The current date is calculated forward by the entered number of months (1 to 65000).
(Adding the letter M to the integer value changes the offset from day to month!)

TIME = Time:
Times are marked with the control character (^).

= ^z 1/100 seconds (not with group H: "^z" always leads to "00" printout)
= ^s seconds
= ^m minutes
= ^h hours
= ^D day of the month
= ^d (Only group H) day of the year
= ^W (Only group H) 3-figure day (e.g. 001 for the 1st day of the year)
= ^M month
= ^Y year, two digit
= ^R (Group H only) year, four digit
= ^G (Group H only) 3-figure month, German
= ^E (Group H only) 3-figure month, English
= ^S (Group H only) Name of a month from command #DM
= ^C (Group H only) Week of the year always with two digits
= ^c (Group H only) Week of the year without leading 0
= ^K (Group H only) Corresponding year to the week (of the year) with 4 digits
= ^k (Group H only) Corresponding year to the week (of the year) with 2 digits

TEXT = Text:
Required ASCII characters are inserted without control characters (^) in the TIME+ text string for printing.

➡ (Only group H) The text field may contain an *input field*.

❍ Syntax description: See chapter Input Fields 🗋 on page 4.

#G = The #YC command must be closed with #G.

Example 1

```
#G -------------------------------------------------------------------------------------------
#G The part of the text which overlaps the line is printed in white, the rest
#G in black.
#G -------------------------------------------------------------------------------------------
#!A1
```

```
#ER
#T5#J5#YL0/0/2.5/90#G
#T5#J5#YC100/0E//Example#G
#Q1/
```

Example 2

```
#G -----------------------------------------------------------------------------------------
#G Printout which is updated during the current print job:
#G Time 08:22.18,07 Date 23-Nov-1998
#G -----------------------------------------------------------------------------------------
#YC105/0U// Time ^h:^m.^s^z Date ^D-^G-^R#G
```

Example 3

```
#G -----------------------------------------------------------------------------------------
#G If the current date is 01/31/2001, the following command generates
#G the printing "02/28/2001".
#G -----------------------------------------------------------------------------------------
#YC109/0/M1/^M/^D/^R#G
```

Example 4

```
#G -----------------------------------------------------------------------------------------
#G If the current date is 09/13/2001, the following command generates
#G the printing "Day of the year : 256".
#G -----------------------------------------------------------------------------------------
#YC109/0/Day of the year : ^d#G
```

#YG Print graphics

Command for printing graphics. It provides the following features:

- Supported graphics formats: BMP, PCX, JPG, TIF, GIF
- Black and white as well as coloured graphics can be used.

➠ Coloured and gray scaled graphics are converted automatically to black and white, using an error diffusion algorithm. This may be very consuming regarding both, the processor capacity as well as the memory capacity. Therefore, it is strongly recommended to use black and white graphics.

Syntax #YG/djg/vo/a/filename#G Group H

d = 0 rotating direction 0°
= 1 rotating direction 90°
= 2 rotating direction 180°
= 3 rotating direction 270°

j = M centered: The writing command (#T/#J) refers to the center of the graphics field. The field is printed on both sides of the print position.

= R Right justified: The writing command (#T/#J) refers to the right margin of the text field. The field will be printed to the left side.

g = D Graphics field consists of a variable field. Definition equals #YT command.

v = + Increment – the offset is added to the text field.
= – Decrement – the offset is subtracted from the text field.

o = int Offset, which is added to (incremented) or subtracted from (decremented) the file name – depending on the polarity.

a = int No. of labels with constant number (1 to 255)

file = File name (including drive letter and path in capital letters) of the graphics file which is ought to be printed. If no path is indicated, the file will be searched in C:\graphics (on the compactflash card!).

❍ For details about admissible file names read topic section "Notes, Definitions, Command Overview", chapter "Number string and text definition" / "File name conventions" and "Drive Names".

Example	#YG/2///EXAMPLE.BMP	Graphics turned 180°. The file must be stored in the "C:\graphics" directory (on the compactflash card).

➠ Memory area for loading and converting of graphics files can be allocated using the parameter SYSTEM PARAMETERS > Free store size.

❍ For information on the current memory partitioning refer to info-printout "Memory status".

#YI Write logo in EPT format directly into the image buffer

Write logo using the **E**asy **P**lug **T**ransfer Format directly to the image buffer.

Definition Command #YI writes the following data directly into the image buffer in a predefined position.

➡ As the logo information is only stored in the image buffer, it will be lost if shifted beyond any layout border.

➡ Only for groups D, E, G:
Status printouts or print checks may not be started during the print job.

➡ Group H: With firmware x.33 or higher, this command is also supported for multi web layouts.

➡ Command must stand between #ER and #Q!

Syntax #YI/s/s/s ... /s#G Groups D, E, G, H

s = hex Coding of a dot line in the logo matrix, hexadecimal with four dots respectively from left to right.

Assignment for hexadecimal coding:

1 = dot is printed
0 = dot is not printed.

Non-set dots at the end of the line can be left out.

Coding of each dot line in the logo matrix using an s-parameter.
Sequence: from the bottom (Line 1) to the top.

➡ Only capital letters and numbers may be used for hex. coding

❍ Notes about creating a logo can be found in the section **Fehler! Verweisquelle konnte nicht gefunden werden.** 🗋.

#G = The command #YI must be closed with #G.

Example

```
#G ------------------------------------------------------------------------------------------
#G Logo with the following construction is written directly into the image
#G buffer.
#G Line 4: 1111 1111 1111           (/FFF)
#G Line 3: 1111 0000 1111 11        (/F0FB)
#G Line 2: 1110 0000                (/E0)
#G Line 1: 1100 0000 0011 0101      (/C035)
#G ------------------------------------------------------------------------------------------
#YI/C035/E0/F0FB/FFF#G
```

#YIB Write logo with binary data directly into image buffer

.

Definition

Command #YIB writes the following binary data directly into the image buffer in a predefined position. The data is represented directly by the 256 ASCII characters, i.e. 8 dots correspond to a byte which is to be transmitted.

➨ As the logo information is only stored in the image buffer, it will be lost if shifted beyond any layout border.

➨ Only for group G:
Status printouts or print checks may not be started during the print job.

➨ Group H: With firmware x.33 or higher, this command is also supported for multi web layouts.

➨ Command must stand between #ER and #Q!

Syntax	#YIBc/d/bbb...b	Groups G, H

c = int Number of dot lines in the logo matrix. Maximum number of lines: 65535.

d = int Number of bytes per line. Maximum: 65535.

➨ All the lines must be of the same length.

b = bin Parameter for one byte. Each byte encodes eight dots.

Assignment for binary coding:
1 = dot is printed, 0 = dot is not printed.
All the lines must be of the same length.

❍ Notes about creating a logo can be found in the section
Fehler! Verweisquelle konnte nicht gefunden werden. 🗋.

Example

```
#G ------------------------------------------------------------------------------------------
#G This command is used to write 3 lines each with 4 bytes
#G (corresponding to 4 * 8 = 32 dots per line) directly into the image
#G buffer.
#G Line 3:     01111100     10011100     10111100     11011100
#G Line 2:     01111011     10011011     10111011     11011011
#G Line 1:     01111010     10011010     10111010     11011010
#G ------------------------------------------------------------------------------------------
#YIB3/4/zÜ¦+{ø+_|£+_
```

#YIR Write logo in RLE format directly into image buffer

Definition

Command #YIR writes the following data directly into the image buffer in a predefined position. The data is coded in RLE Format (Run Length Format).

➡ As the logo information is only stored in the image buffer, it will be lost if shifted beyond any layout border.

➡ Only for group G:
Status printouts or print checks may not be started during the print job.

➡ Group H: With firmware x.33 or higher, this command is also supported for multi web layouts.

➡ Command must stand between #ER and #Q!

Syntax #YIRc/sss...s Group G, H

c = int Number of dot lines in the logo matrix. Maximum number of lines: 65535.

s = hex Coding of a dot line in the logo matrix.

The parameter s consists of a sequence of hexadecimal digits (hex numbers). Permitted value range for each of the hex numbers: 00 to FD. The values FE and FF are reserved for control characters as described below.

Start-Code

The start code must stand at the beginning of the data for a dot line. A dot line which is to be printed several times in succession only needs to be entered once with the start code FF. The number after FF shows how often the line is to be repeated. Example: the start code FF04 produces four identical lines.

FE (hex):	start code for lines which are to be printed once.
FF (hex):	start code for lines which are to be printed repeatedly.

End-Code

The last dot line in a logo matrix must be closed with the end code FE.

RLE-Format

The decisive factor when converting binary code into RLE format is the changing from ones to zeros and vice-versa. In order to convert binary code into RLE format, it is only necessary to count the sequential zeros and ones between the start and end code and to note them as hex numbers. Single figure hex numbers are prefixed with a zero.

Example: Binary code: 0001111001111111
RLE code: FE03040207FE
(start code FE, 03 zeros, 04 ones, 02 zeros, 07 ones, end code FE if no other dot lines follow)

➡ To enter the Code in RLE-Format as shown in the example, you need to a) use an editor providing a hex-mode option and b) enable the hex-mode. Using an editor without hex-mode would require to enter the corresponding ASCII-character to every hex number, for example ♥ instead of 03.

➡➡

Dot-Line

The actual data of a dot line is contained in pairs of hex numbers between the start and end code. The first hex number of the pair gives the number of consecutive zeros, the second hex number gives the number of consecutive ones.

Example: 05FC (05 zeros, 252 ones)

If the number of zeros or ones in the binary code exceeds 253, a new pair of hex numbers must be started (the correspondences of the numbers 254 and 255 are the start codes FE and FF).

Example: Binary code: 262 ones, 3 zeros, 1 one
RLE code: FE00FD00090301
(start code FE, 00 zeros, FD ones, 00 zeros, 09 ones, 03 zeros, 01 ones)

The 262 ones are divided into two value pairs because there are more than 253: 00FD and 0009. As the binary code does not have any zeros at the beginning, the first byte in the first and second value pair is a zero.

Line length

RLE lines can be of any length.

Blank lines

A single blank line has the form FE0000.
Several blank lines are represented as follows: start code FF, followed by the number of required blank lines.

Example: FF04 (04 blank lines in sequence); FF01 = FE0000 (one blank line)

➡ No zeros are required after FFxx (with xx blank lines). The start code for the next line can follow directly afterwards.

❍ Notes about creating a logo can be found in the section **Fehler! Verweisquelle konnte nicht gefunden werden.** 🗋.

Example

```
#G ------------------------------------------------------------------------------------------
#G With this command 7 lines are written directly into the image buffer.
#G Line 7: 00111000011111
#G Line 6: 00111000011111
#G Line 5: 00111000011111
#G Line 4: 00111000011111
#G Line 3: blank line
#G Line 2: 111...(262 times)...1110001
#G Line 1: 0001111001111111
#G ------------------------------------------------------------------------------------------
#YIR7/FE03040207FE00FD00090301FE0000FF0402030405FE
```

#YK Logo definition

Definition

This command is used to print a logo. The logo must be loaded in the memory first.

The logo is a data block and is positioned in the label according to a predefined reference point (#T and #J command).

➡ Command must stand between #ER and #Q!

Syntax	#YKn/d	Groups A, B
	#YKn/dj	Groups D, E, G
	#YKn/djm	Group H

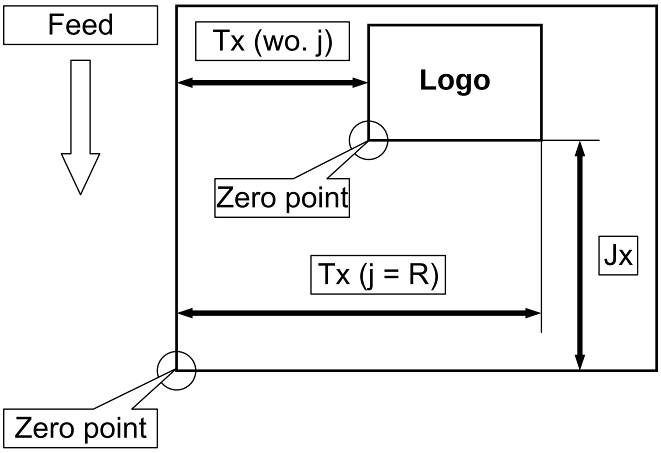
n = int Logo number (0 to 255)

d = 0 Logo in normal write direction
= 1 Logo rotated by 90 degrees
= 2 Logo rotated by 180 degrees
= 3 Logo rotated by 270 degrees

j = M Middle:
The command #T refers to the centre of the field. The logo is hereby created on both sides.

= R Flush right:
The command #T refers to the right edge of the field. The logo is built up to the left.

m = The same as m = A.
= A The logo is copied to the internal RAM disc (default setting).
= C The logo is copied to CompactFlash card.
(in directory "\logos", respectively)



Example	#YK10/0	Logo number 10 is printed, if available.

#YL Line definition

Definition This command is used to define lines of varying lengths and thicknesses.

➡ Command must stand between #ER and #Q!

➡ If lines are required as a background, they must be defined before the foreground.

Syntax		
	#YLa/d/h/l	Groups A, B
	#YLa/drz/h/l	Groups D, E, G, H

a = int Line type number.

❍ Print a copy of the info-printout "Font status" (group H) or "STA2" (other groups) respectively, to get a sample of the available line types and numbers. For detailed information on info-printouts refer to topic section "Info-printouts and parameters".

d = 0 Line in normal write direction
= 1 Line rotated by 90 degrees
= 2 Line rotated by 180 degrees
= 3 Line rotated by 270 degrees

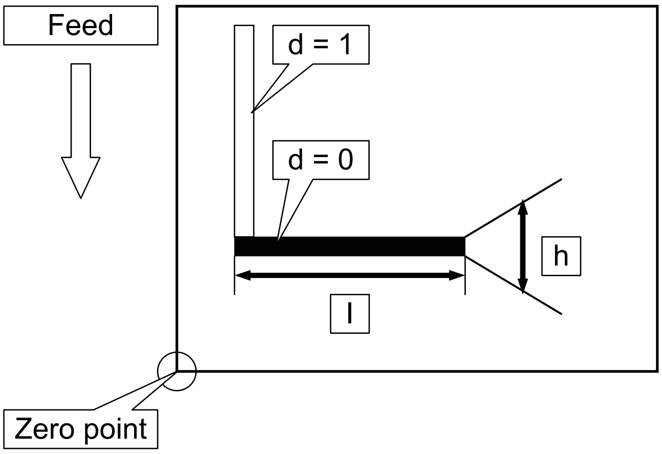
r = R The line is recreated for each label (e.g. as the background of a number field).

➡ Don´t use parameter r together with inverse fields! – otherwise the field contents will be overprinted by the line.

z = P Normal black printout
= A “White printout”, that is the printing is left blank. Requires a dark background
= E Printout with inverted bitimage (black is left blank; white is printed black)

h = num Line thickness in mm

l = num Line length in mm



Example	#YL1/0/3/20	Line type 1, 0 degree rotation, 3 mm thick, 20 mm long
	#YL4/3/1/50	Line type 4, 270 degree rotation, 1 mm thick, 50 mm long

#YN Text field

Definition

Printing text with *scalable fonts*.

❍ Non-scalable fonts (fixfonts) are also available and can be printed with the command #YT.

The command #YN defines the size and font of a text, as well as the direction of rotation and consecutive numbering.

The following must be taken into consideration when using consecutive numbers:
- A sufficient number of positions must be defined corresponding to the biggest number to be used. Leading zeros are printed.
- Only numerical characters are incremented or decremented.
- All numerical characters within a text string are incremented or decremented.

➡ Command must stand between #ER and #Q!

Syntax		
	#YNz/dbjf/e/vo/a/TEXT#G	Groups D, E, G
	#YNz/dbqmjwcghrkp/e/vo/a/TEXT#G	Group H

z = int Font number

➡ The fonts 100 and 101 are internal scalable fonts and are therefore always available.

❍ See user manual of the relevant printer type, topic section Internal Fonts 🗋.

➡ Non-internal fonts can be loaded from Compact Flash card.

❍ For detailed information refer to the manual "Plug-in cards", topic section "Application", paragraph CF/SD cards 🗋.

d = 0 Normal write direction
= 1 Text rotated by 90 degrees
= 2 Text rotated by 180 degrees
= 3 Text rotated by 270 degrees

b = A Text printed inversely:
A dark background must first be applied to the print area (line or rectangle).
Inverse printed offset is not possible within Groups A and B.

q = E Existing Bitimage is inverted (see example 1).

m = X Text field consists of one variable data field

n = Y Suppression of leading zeros in counting fields. The remaining digits are printed at the same position as with leading zeros (see example 2).

j = M Middle:
The write command (#T/#J) refers to the centre of the text field. The field is built up on both sides of the print position.

= R Flush right:
The write command (#T/#J) refers to the right edge of the text field. The field is built up to the left.

f = F Field designation for changing data
(see description of changing data)

w = W Counter field TEXT is incremented/decremented without a carry over, i.e. only the first unit position of the figure is increased/decreased.

c = Dint Rotation of TEXT in steps of one degree (0 to 359 degree);
Default setting: 0 degree; The character D has to be succeeded without blank by the value. Parameter c takes priority over d.

g = O Default type
= B Bold

h = V Default type
= I Italics

r = The same as "K"
= K Application of scalable fonts without Unicode (default setting)
= U Application of Unicode fonts.
= G Application of arabic Unicode fonts (characters will be printed right to left)

Each Unicode-character has to be referred to in the TEXT-field as follows: *\u0041* (example with 0041 being the index of the character "A").

❍ See example 2.

➠ There are some specific features to bear in mind, when using Unicode fonts!

❍ See topic section "General Notes, Definitions and Command Overview", chapter Easy Plug and Unicode 🗋.

k = Snum Fixed distance in mm between all characters in TEXT. The character S has to be succeeded without blank by the value.

p = N (Default) Standard character-algorithm is applied.
Q 2 D character algorithm is used. May improve the exactness of small characters. (e.g. printing of small chinese characters). This algorithm is about 30-40% slower than the standard algorithm.

e = int Font size in Dot, depending on the font and its installed size, respectively.

Group	Font size / Dot
C	15 to 2000
E, G	15 to 3000
H	5 to 6000
other groups	15 to 300

➠ Group H only:
Optionally can be defined different font sizes for x- and y-direction (see example 3).

v = + Incrementation – offset is added to the text field.
= - Decrementation – offset is subtracted from the text field.

o = int Offset, which is added to (incremented) or subtracted from (decremented) the TEXT depending on the prefix.

a = int Number of labels with the same constant number (1 to 255).

TEXT = Any alphanumerical text, whereby only numerical values are taken into account during offset. All numerical values of the text string are taken into account for the offset (e.g. 10A3B56 = 10356). Max. number of characters: 255.

➡ The text field may also be a *variable data field*. Precondition: the X-flag is set.

➡ Variable data fields must not be used with Unicode!

❍ See chapter Variable Data Fields on page 3.

➡ (Only group H) The text field may contain an *input field*.

❍ See chapter Input Fields on page 4.

#G = The command #YN must be closed with #G.

Example 1
Inverted Bitimage

```
#!A1
#ER
#T5#J5#YL0/0/2.5/90#G
#G ------------------------------------------------------------------------------
#G The part of the text which overlaps the line is printed in white, the rest
#G in black.
#G ------------------------------------------------------------------------------
#T5#J5#YN100/0E////Example#G
#Q1/
```

Example 2
Suppressing leading zeros

```
#G ------------------------------------------------------------------------------
#G Printout: Test  1, Test  2, …, Test 10, etc.
#G ------------------------------------------------------------------------------
#YN100/0Y/1/1//Test000#G
```

Example 3
Scaling speedo fonts

```
#G ------------------------------------------------------------------------------
#G The text „Speedo Test“ has font size 120 in y-direction and
#G font size 99 in x-direction.
#G ------------------------------------------------------------------------------
#YN100/0/120X99///Speedo Test#G
```

Example 4
Scaling speedo fonts

```
#G ------------------------------------------------------------------------------
#G Font 100 in size 10 points.  The 4-figure number is incremented on
#G every 5th label by the figure "1" (5 x 0001, 5 x 0002).
#G ------------------------------------------------------------------------------
#YN100/0/10/1/5/0001#G
```

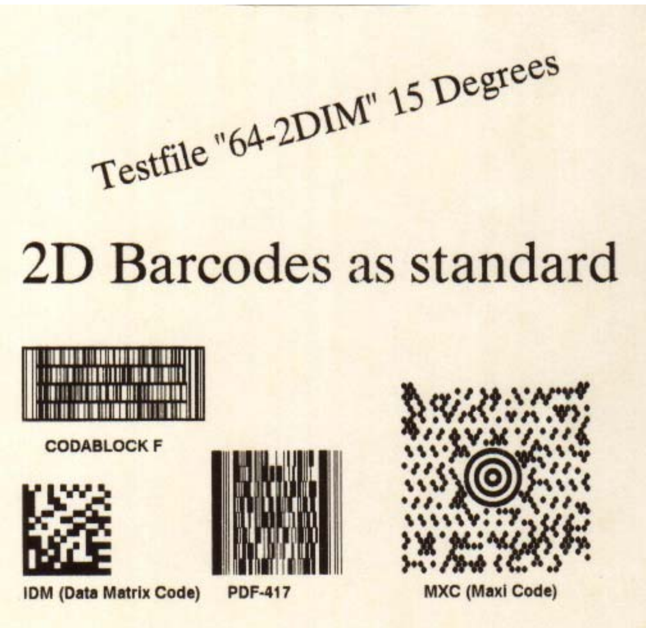
➽

Example 5
Printing Unicode characters

```
#G ------------------------------------------------------------------------------------------
#G Printout:
#G „Example printout of the character „A“ using a Unicode font: A”
#G ------------------------------------------------------------------------------------------
#!A1
#G --------------------------------------------
#G Endles material, 100 mm wide, 30 mm long
#G --------------------------------------------
#IMN100/30
#ERN
#T5#J10
#G --------------------------------------------
#G Using font 902; printout in normal writing direction; Unicode;
#G Font  size 60 dots („\u0041“ is the Unicode for „A“)
#G --------------------------------------------
#YN902/0U/60///Example printout of the character „A“ using a Unicode
font:  \u0041#G
#Q1#G
```

Example 6
Printout: see Fig. 7

```
#G ------------------------------------------------------------------------------------------
#G This  Easy Plug program generates the printing pictured in Fig. 7.
#G The printing includes four 2dimensional bar codes and a
#G 35-degrees-rotation.
#G ------------------------------------------------------------------------------------------
#!A1
#G --------------------------------------------
#G Punched material, 100 mm wide, 80 mm long
#G --------------------------------------------
#IMS100/80
#ER
#J60#T15#YN100/0D15/60///Testfile "64-2DIM" 15 Degrees#G
#J48#T5#YN100/0/90///2D Barcodes as standard#G
#J10#T55#MXC4/0/2/3///1234567890abcdefghijklmnopqrstuivwxyz#G
#J10#T30#PDF0/T0/1/0/0/2/2/1234567890ABCDEFG#G
#J10#T5#IDM/0/10///1234567890abc#G
#J7#T5
#YT105/0///IDM (Data Matrix Code)   PDF-417   MXC (Maxi Code)#G
#J30#T5
#CBF/0/2/2/8/0///CODABLOCK F 3456789012345678901004 0digit#G
#J26#T8#YT105/0///CODABLOCK F#G
#Q1/
```



Testfile "64-2DIM" 15 Degrees

2D Barcodes as standard

Fig. 7 Printout belonging to example 6.

Example 7
Printout: see Fig. 8

```
#G -------------------------------------------------------------------------------------------
#G Prints the chinese character (Fig. 8) with the Unicode u50D1.
#G -------------------------------------------------------------------------------------------
#!A1
#IMN100/100/
#ERN
#T10#J10
#YN902/0U/400///\u50D1#G
#G -------------------------------------------
#G „\u50D1“ is the Unicode of the printed character
#G -------------------------------------------
#Q1#G
```

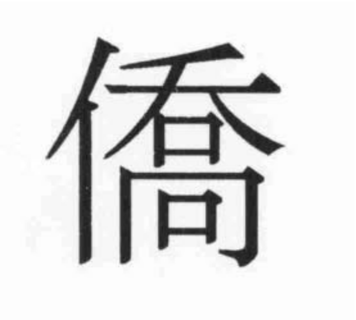


Fig. 8 Printout belonging to example 7.

#YR Rectangle definition

Definition This command is used to define rectangles of varying sizes and line thicknesses.

➨ Command must stand between #ER and #Q!

Syntax	#YRa/d/h/l/b #YRa/dr/h/l/b	Groups A, B Groups D, E, G, H

a = int Line type number.

❍ Print a copy of the info-printout "Font status" (group H) or "STA2" (other groups) respectively, to get a sample of the available line types and numbers. For detailed information on info-printouts refer to topic section "Info-printouts and parameters".

d = 0 Rectangle in normal write direction
= 1 Rectangle rotated by 90 degrees
= 2 Rectangle rotated by 180 degrees
= 3 Rectangle rotated by 270 degrees

The rectangle rotates around the reference point.

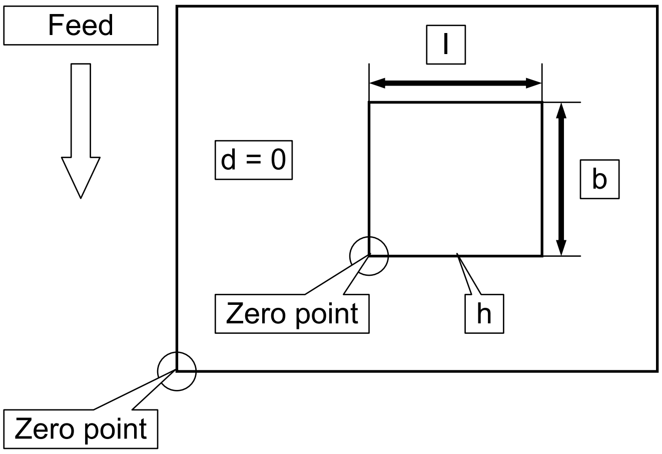
r = R The rectangle is recreated for each label (e.g. as the background of a number field).

➨ If rectangles are required as a background, they must be defined before the foreground.

h = num Line thickness in mm

l = num Width of the rectangle in mm

b = num Height of the rectangle in mm



Example	#YR0/0/2/15/25	Rectangle type 0, 0 degree rotation, line thickness 2 mm, 15 mm wide, 25 mm high.

#YS Real time as bar code

Definition

A time in text format can be defined in the form of a bar code using this command. The printer must be equipped with a real-time clock. Normal text for the plain-copy line can also be entered in conjunction with the time.

(^^), (##) in the string: control characters (^) and hashes (#) are printed as bar code (only in the case of alphanumerical bar codes).

➡ Command must stand between #ER and #Q!

Not with group H:
Any number of #YC commands are possible in the format. However, per print job only up to 3 commands (#YC or #YS) can be assigned the option U (update during the print job).

Syntax		
	#YSz/dkcu/h/s/o/TIMETEXT#G	Group B
	#YSz/dkcfu/h/s/o/TIMETEXT#G	Groups E, G, TTX 207/Da Capo
	#YSz/dkbcujlrm/h/s/o/TIMETEXT#G	Group H without AP 4.4

z = int Bar code number (see Bar Code Library)

d = 0 Bar code in normal write direction
= 1 Bar code rotated by 90 degrees
= 2 Bar code rotated by 180 degrees
= 3 Bar code rotated by 270 degrees

k = M Bar code with plain-copy line
= O Bar code without plain-copy line

b = A Position change of the plain-copy line. If the plain-copy line is normally printed below the bar code, it is shifted by this option to the top of the bar code, and the other way round.

c = C A check digit according to Module 10 is calculated and printed.

u = (void) as u = W.

= W (Default setting) No update. The time is read into the variable *once*. This is done during the compilation of the layout.
= U The time is read repeatedly during the running print job. This happens immediately after printing the relevant output field (#VW…), but counts only for the same field on the *next* label.

➡ The setting is not suitable for dispensers.
Reason: If the next label is printed considerably later (e.g. in single-start mode with footswitch), the following happens: The output field on label 2 contains the time, at which the same field on label 1 was printed.
Example: Label 1 is printed at 10.50 h, label 2 at 11.03 h. In this case, the imprint of label 2 shows the time 10.50!

= Y (Only group H - recommended for dispensers) The time is read repeatedly during the printing of a label (approx. all 5 seconds).

j = Z Field position centered, related to the print position.
= R Field position right-justified, related to the print position. The field is built up starting at the print position to the left side.
= (Default setting) Field position left-justified, related to the print position. The field is built up starting at the print position to the right side.

l = G Plain copy line default position

= H Plain copy line justified

= I Plain copy line left-justified

= K Plain copy line centered

= L Plain copy line right-justified

r = Pnum Ratio of the bar code (2.0 to 3.0)
The default setting is 2.5. The letter P (proportion) must stand immediately in front of the ratio (e.g. P2.5). A ratio without the letter P is invalid.

➡ Observe the sequence: The ratio must be entered after the parameter d (write direction)!

f = F (Only group E) Field designation for exchanging data
(see description of exchanging data)

m = B EAN/UCC mode with brackets around the data designator.
The data have to be sent in brackets! The brackets appear in the plain copy line but not as bar code.

= X EAN/UCC mode without brackets around the data designator.
Data have to be sent without brackets.

h = int Bar code height:
0 = 5 mm, 1 = 10 mm, ... from 10 = actual height in mm
(e. g. 35 = 35 mm high). This value minus 1 is multiplied by the value in parameter BCHI set at the printer.

With the printers TTX 207/300/480, Mustang 900/1000, Cobra and S 20/30, the value entered corresponds to the effective bar code height
(0 = 1 mm, 1 = 2 mm, 3 = 3 mm etc.).

s = int Bar code width (1 to 16)

o = (blank) The current date is printed.

= 0 The current date is printed.

= int Days offset: The current date is calculated forward by the entered number of days (1 to 65000).

= Mint (Only for group H) Months offset: The current date is calculated forward by the entered number of months (1 to 65000).
(Adding the letter M to the integer value changes the offset from day to month!)

TIMETEXT = String containing Time and Text (ASCII-characters) in any order. Times are marked with the control character (^). Insert required Text as ASCII characters (without control characters) in the TIMETEXT-string.

Time formats:

= ^ z 1/100 seconds (not with group H: "^ z" always leads to "00" printout)

= ^ s Seconds

= ^ m Minutes

= ^ h Hours

= ^ D Day of the month

= ^ d (Only group H) day of the year

= ^ W (Only group H) 3-figure day (e.g. 001 for the 1st day of the year)

= ^ M Month

= ^ Y Year, two digit

	=	^ R	Year, four digit
	=	^ G	3-figure month, German
	=	^ E	3-figure month, English
	=	^ S	Name of a month from command #DM
	=	^ C	(Only group H) Week of the year always with two digits
	=	^ c	(Only group H) Week of the year without leading 0
	=	^ K	(Only group H) Corresponding year to the week (of the year) with 4 digits
	=	^ k	(Only group H) Corresponding year to the week (of the year) with 2 digits
#G	=		The #YS command must be closed with #G.

Example	#YS13/0MU/2/2//^ D-^ G-^ R#G	Printout which is updated during the current print job: Code 128 with plain-copy line "23-Nov-1998", normal print direction, 10 mm high, width 2

#YT Text field definition

Definition This command is used for (optional) parameter settings and entering text.

➡ Command must stand between #ER and #Q!

Syntax	#YTz/dbj/vc/a/TEXT	Group A
	#YTz/dbjf/vo/a/TEXT	TTX 207/Da Capo
	#YTz/dtbj/vo/a/TEXT	Group B
	#YTz/dbfgj/vo/a/TEXT	Groups E, G, TTX 300/Cobra Online
	#YTz/dbqgnwjk/vo/a/TEXT	Group H

z = int Font number (YT100 to YT116)

❍ Refer to the Info-Printout PRINT INFO > Font status for group H or INFO > STA2 for all other groups or refer to the user manual, topic section "Internal Fonts" for a listing of all internal fonts.

➡ In case of an invalid font number, the text is printed with YT100.

d	=	0	Normal write direction
	=	1	Text rotated by 90 degrees
	=	2	Text rotated by 180 degrees
	=	3	Text rotated by 270 degrees
t	=	N	Text printed normally
	=	K	Text printed in italics
b	=	A	Text printed inversely: A dark background must first be applied to the print area (line or rectangle). Inverse printed offset is not possible within Groups A and B.
q	=	P	Normal black printout
	=	A	“White printout”, that is the printing is left blank. Requires a dark background
	=	E	Printout with inverted bitimage (black is left blank; white is printed black)

❍ See example 1.

j = M Middle:
The write command (#T / #J) refers to the centre of the text field. The field is built up on both sides of the print position.

= R Flush right:
The write command (#T / #J) refers to the right edge of the text field. The field is built up to the left.

k = Snum Fixed distance in mm between all characters in TEXT. The character S has to be succeeded without blank by the value.

f = F Field designation for changing data
(see description of changing data)

g = D Text field consists of one variable data field

n = Y Suppresses leading zeros in counter fields. The remaining digits are printed at the same position they were printed at *with* leading zeros.

❍ See example 2.

w = W Counter field TEXT is incremented/decremented without a carry over, i.e. only the first unit position of the figure is increased/decreased.

v = + Incrementation – offset is added to the text field.
= - Decrementation – offset is subtracted from the text field.

➡ If the text field consists of a variable data field, incrementation/decrementation is *not* possible.

o = int Offset, which is added to (incremented) or subtracted from (decremented) the TEXT depending on the prefix.

a = int Number of labels with the same constant number (1 to 255).

TEXT = Any alphanumerical text, whereby only numerical values are taken into account during offset. All numerical values in the text string are taken into account for the offset (e.g. 10A3B56 = 10356). Maximum number of characters: 255.

➡ The text field may also be a *variable data field*. Precondition: the D-flag must be set.

❍ Syntax description: See chapter Variable Data Fields 🗋 on page 3.

➡ (Only group H) The text field may contain an *input field*.

❍ Syntax description: See chapter Input Fields 🗋 on page 4.

#G The command #YT must be closed with #G.

Example 1
Inverted Bitimage

```
#G ------------------------------------------------------------------------------------------
#G The part of the text which overlaps the line is printed in white, the rest
#G in black.
#G ------------------------------------------------------------------------------------------
#!A1
#ER
#T5#J5#YL0/0/2.5/90#G
#T5#J5#YT100/0E///Example#G
#Q1/
```

Example 2

```
#G ----------------------------------------------------------------------------------------------------
#G Printout: Test  1, Test  2, …, Test 10, etc.
#G ----------------------------------------------------------------------------------------------------
#YT106/0Y/1/1/Test000#G
```

Example 3

```
#G ----------------------------------------------------------------------------------------------------
#G Font 100, without parameter.
#G ----------------------------------------------------------------------------------------------------
#YT100/0///DEMO#G
```

Example 4

```
#G ----------------------------------------------------------------------------------------------------
#G Font 101, italic text and rotated by 180 degrees.
#G ----------------------------------------------------------------------------------------------------
#YT101/2K///DEMO#G
```

Example 5

```
#G ----------------------------------------------------------------------------------------------------
#G Font 100, text (10) is increased by 1 every 5 labels (DEMO-10,
#G DEMO-11, DEMO-12 etc.).
#G ----------------------------------------------------------------------------------------------------
#YT100/0/1/5/DEMO-10#G
```

Example 6

```
#G ----------------------------------------------------------------------------------------------------
#G Printout: Text
#G ----------------------------------------------------------------------------------------------------
#YT109/0///T#G#YT103/0///ext#G
```

Example 7

```
#G ----------------------------------------------------------------------------------------------------
#G Font 104 with variable data field   Number 00 with 15 characters.
#G ----------------------------------------------------------------------------------------------------
#YT104/0D///$00,15#G
```

#YV Variable data field

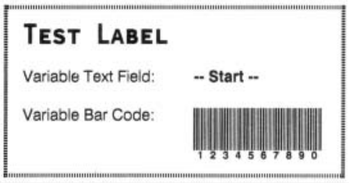
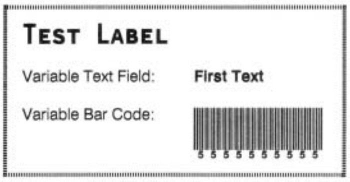
Definition This command is used for filling a field with variable data. Field types for variable data are text fields and bar code fields. The variable data field must be made known to the printer before the command #YV is used (see #YT, #YB).

➡ To draw a maximum benefit of the printers performance, print jobs should send the variable data for the #YV field in the same order in which the fields are printed. Thus, the fields can be updated while printing. Otherwise, the printer has to wait with updating, until the respective #YV field is printed, what can slow down the throughput!

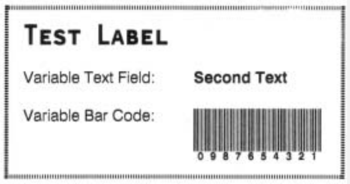
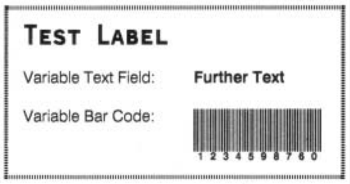
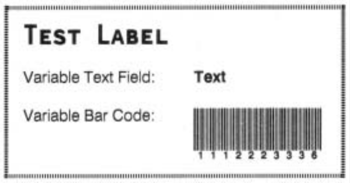
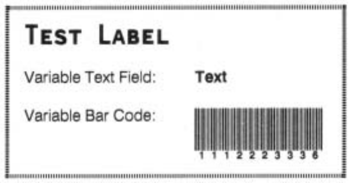
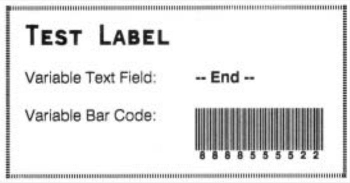
➡ Variable data fieds may not be combined with counting fields!

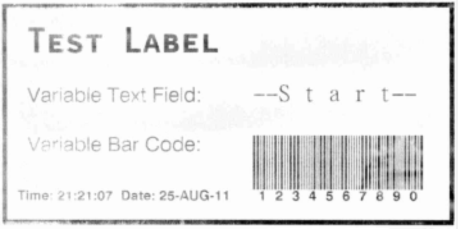
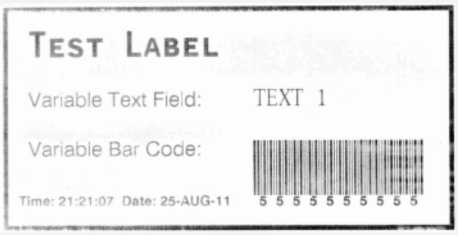
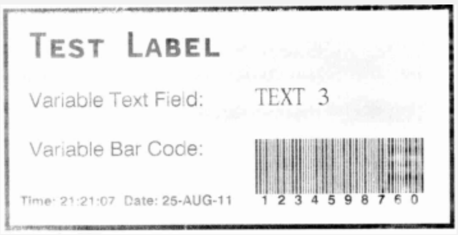
Syntax #YVzn/TEXT#G Groups E, G, H

z =	int	Data field number (0 to 99); (Gruppe H: 0 to 999)
n =		Trailing blanks in TEXT are deleted.
=	B	Trailing blanks in TEXT are *not* deleted.
TEXT =		Variable data (1 to 255 characters)
#G =		The command #YV must be closed with #G.

Example (using #YT)		#!A1 #IMS95/50 #ERN #T5#J5 #YR3//1/90/45 #T10#J40 #YT109////Test Label#G #T10#J30 #YT104////Variable Text Field:#G #T10#J20 #YT104////Variable Bar Code:#G #T55#J30 #YT107/D0///$00,15 #T55#J10 #YB6/D0/10/3///$01,10
	 TEST LABEL Variable Text Field: -- Start -- Variable Bar Code: 1 2 3 4 5 6 7 8 9 0	#YV00/-- Start --#G #YV01/1234567890#G #Q1#G
	 TEST LABEL Variable Text Field: First Text Variable Bar Code: 5 5 5 5 5 5 5 5 5 5	#YV00/First Text#G #YV01/5555555555#G #Q1#G

A = TTX 600 --TLX 604 – Eagle **B** = TTX 300 Offline - Cobra Offline **D** = TTX 300 Online –TTX 207 - Cobra Online – Da Capo
E = TTX 450/650/674/675/950/1050 – Puma/Puma plus/Lion/Lion plus/Tiger/Tiger XXL -- S45/65/95/105 - TDI/STDI/XXTREME
G = TTX 350 – Ocelot – TTK – Texxtile **H** = 64-xx – Chess x – DPM – PEM – ALX 92x – AP 4.4 – AP 5.4 – AP 5.6 – AP 7.t

	 TEST LABEL Variable Text Field: Second Text Variable Bar Code: 0987654321	#YV00/Second Text#G #YV01/0987654321#G #Q1#G
	 TEST LABEL Variable Text Field: Further Text Variable Bar Code: 1234598760	#YV00/Further Text#G #YV01/1234598760#G #Q1#G
	 TEST LABEL Variable Text Field: Text Variable Bar Code: 1112223336  TEST LABEL Variable Text Field: Text Variable Bar Code: 1112223336	#YV00/Text#G #YV01/1112223336#G #Q2#G
	 TEST LABEL Variable Text Field: -- End -- Variable Bar Code: 8888555522	#YV00/-- End --#G #YV01/8888555522#G #Q1#G
Example (using #YN)		#!A1 #IMN95/50 #ER #T5 #J5 #YR0//1/90/45 #T8#J10 #YC106/0//Time: ^h:^m:^s Date: ^D-^G-^R#G #T10#J40 #YT109////Test Label#G #T10#J30 #YT104////Variable Text Field:#G #T10#J20 #YT104////Variable Bar Code:#G #T55#J30 #YN902/0X/60///$00,15 #T55#J10 #YB6/D0/10/3///$01,10

	 TEST LABEL Variable Text Field: --S t a r t-- Variable Bar Code: Time: 21:21:07 Date: 25-AUG-11 1 2 3 4 5 6 7 8 9 0	**#YV00/--S t a r t--#G** #YV01/1234567890#G #Q1#G
		#YV00/TEXT 1#G #YV01/5555555555#G #Q1#G
	 TEST LABEL Variable Text Field: TEXT 1 Variable Bar Code: Time: 21:21:07 Date: 25-AUG-11 5 5 5 5 5 5 5 5 5 5	**#YV00/TEST 2#G** #YV01/0987654321#G #Q1#G
	 TEST LABEL Variable Text Field: TEXT 3 Variable Bar Code: Time: 21:21:07 Date: 25-AUG-11 1 2 3 4 5 9 8 7 6 0	**#YV00/TEXT 3#G** #YV01/1234598760#G #Q1#G

Z

#Z Mirroring

Creating a mirror image

Definition

The command #Z allows mirror images of texts, bar codes, logos, lines and rectangles to be created.

All fields which stand before the command are mirrored.

The mirror axis is the centre axis of the label in relation to the label width given in the command #IM.

Group H:
The label width indicated in the #IM command is rounded up to a multiple of 2.66 mm (32/12 mm). The mirroring axle is the center of the label related to the rounded up label width.

➡ Command must stand between #ER and #Q!

Syntax	#Z	All groups

⚠

➡ Misprints occur with update fields such as number field, clock, line repeat and Easyline!

➡ Group H: If the example printjob is printed repeatedly in standalone mode, only the first printjob will contain mirrored text. The following printjobs will show the text unmirrored.

Example

```
#!A1
#IMN64/100
#ERN
#G -------------------------------------------
#G Mirrored text
#G -------------------------------------------
#T5#J5#YT106////Mirror
#Z
#G -------------------------------------------
#G Normal text
#G -------------------------------------------
#T20#J50#YT107////Normal print
#Q10/
```

Index